Using the BASIC/UX System

HP 9000 Series 300 Computers
BASIC/UX 5.5

HP Part Number 98796-90000

HEWLETT PACKARD

Hewlett-Packard Company
3404 East Harmony Road, Fort Collins, Colorado 80525

Notices

The information contained in this document is subject to change without notice.

Hewlett-Packard makes no warranty of any kind with regard to this manual, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Hewlett-Packard shall not be liable for errors contained herein or direct, indirect, special, incidental or consequential damages in connection with the furnishing, performance, or use of this material.

Warranty. A copy of the specific warranty terms applicable to your Hewlett-Packard product and replacement parts can be obtained from your local Sales and Service Office.

Win an HP Calculator!

Your comments and suggestions help us determine how well we meet your needs. **Returning this card with your name and address enters you into a quarterly drawing for an HP calculator*.**

Using the BASIC/UX System

	Agree				Disagree
The manual is well organized.	○		○	○	○
It is easy to find information in the manual.	○	○	○	○	○
The manual explains features well.	○	○	○	○	○
The manual contains enough examples.	○	○	○	○	○
The examples are appropriate for my needs.	○	○	○	○	○
The manual covers enough topics.	○	○	○	○	○
Overall, the manual meets my expectations.	○	○	○	○	○

You have used this product:

__ Less than 1 week __ Less than 1 year __ More than 2 years
__ Less than 1 month __ 1 to 2 years

fold

Please write additional comments, particularly if you disagree with a statement above. Use additional pages if you wish. The more specific your comments, the more useful they are to us.

Comments: _____

*Offer expires 10/1/1990 (Manual: 98796-90000 E0988)

Please Tape Here

Please print or type your name and address.

Name: _____

Company: _____

Address: _____

City, State, Zip: _____

Telephone: _____

Additional Comments: _____

Using the BASIC/UX System
HP Part Number 98796-90000
E0988

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 37 LOVELAND, COLORADO

POSTAGE WILL BE PAID BY ADDRESSEE

Hewlett-Packard Company
Attn: Learning Products Center
3404 East Harmony Road
Fort Collins, Colorado 80525-9988

Printing History

New editions of this manual incorporate all material updated since the previous edition. Update packages may be issued between editions and contain replacement and additional pages which you merge into the manual. Each updated page has a revision date at the bottom of the page.

The manual printing date and part number indicate its current edition. The printing date changes when a new edition is printed. (Minor corrections and updates incorporated at reprint do not cause this date to change.) The manual part number changes when extensive technical changes are incorporated.

November 1988 ... Edition 1

Contents

1

Important Information

This book describes how to use HP BASIC/UX (called BASIC/UX from here on). You can find information on:

- Entering and leaving BASIC/UX
- Using X Windows (a window system supported on bit-mapped displays)
- Performing tasks with BASIC
- Using HP-UX commands in BASIC/UX.

What this book does *not* do is:

- Describe in detail the differences between BASIC/UX and Workstation BASIC (see *Programming Techniques, Volume 2,* "BASIC/UX Differences and Enhancements")
- Show you how install the system (see *Installing and Maintaining the BASIC/UX System*).

What's the Difference Between BASIC/UX, BASIC and HP-UX?

- BASIC refers to the BASIC commands and programming language system (the functionality of Workstation BASIC).
- HP-UX is Hewlett-Packard's implementation of the UNIX™ operating system. (UNIX™ is a registered trademark of AT&T in the U.S.A. and other countries.)
- BASIC/UX ("BASIC on UNIX") refers to the coordination of HP BASIC with the underlying HP-UX system.

For More Information

To install BASIC/UX, see the *Installing and Maintaining Your BASIC/UX System* manual.

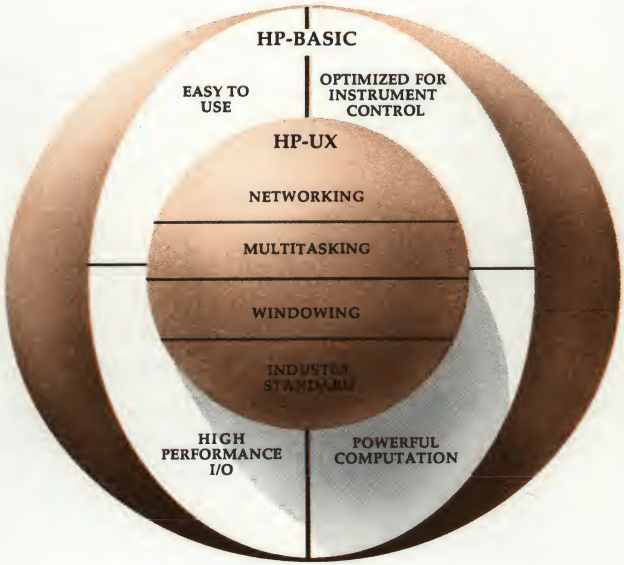


Figure 1–1. HP BASIC/UX Related to HP-UX

How to Recognize Typing Conventions Used in This Book

Table 1–1. Font Conventions

Typing Font	What it means	Example
`literal`	Response from the computer or something you type character for character.	`rmb` [Return] means you type the characters `rmb` and press the [Return] key.
variable	When you see this type of print in a command, it means you must supply a value.	`MSI "`*dir*`"` is a command to change directories. *dir* is a directory name you supply for the command to work, for example `dir1`.
term	Words in this print are new terms.	**multi-tasking**

Things to Know Before Starting

Your Friend: The System Administrator

The system administrator sets up (installs) and manages your system, for example:

- Add disks to the system
- Add new users to the system
- Perform system back-ups.

If *you* happen to be the system administrator, then you should refer to the *Installing and Maintaining the BASIC/UX System* manual whenever this book refers to the system administrator.

Where to Go Next

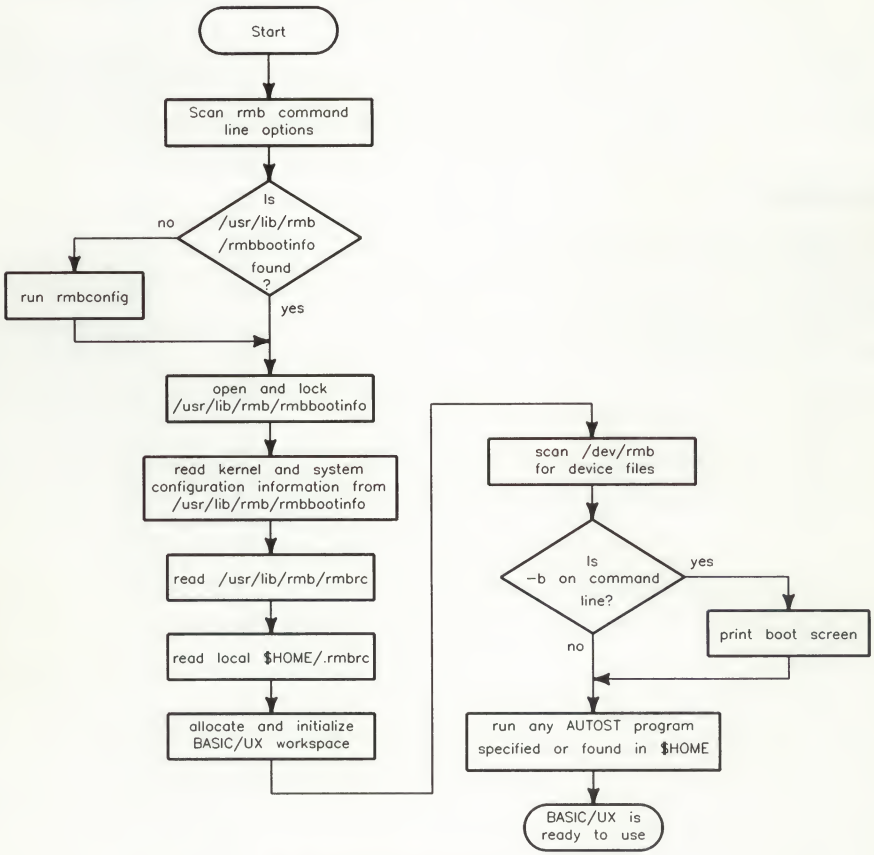
Table 1-2. Where do I go now?

Have you installed BASIC/UX?	Go here ...
No	*Installing and Maintaining the BASIC/UX System*
Yes	Chapter 2

2

Entering and Leaving BASIC/UX

After you install your system (see *Installing and Maintaining the BASIC/UX System*), you are ready to use BASIC/UX.



HP BASIC/UX Boot Process

Signing on to the System (login)

Press Return a few times.

Table 2–1. Check the Status of the Computer

If you see ...	Do this ...
`login:`	Continue with this module.
`$`	(or a similar prompt) you don't need to login. Check if someone else is already logged in, then skip ahead to "Leaving BASIC/UX (logout)."
(nothing)	(i.e. a blank screen) check the power, screen brightness, or see your system administrator.

Prerequisites

See your system administrator for:

- Your user name
- Your password.

Logging in

Follow this example and substitute *your* user name and password for the ones in the example. This example uses:

- user name = `leslie`
- password = `password1`

Table 2–2. How to Login to the System

Do this ...	What you should see	Comments
1. Press Return several times	`login:`	See your system administrator if you don't see this prompt.
2. Type user name	`login: leslie`	Press Break for typing mistakes and try again; it won't accept Back space.
3. Press Return		Enters user name.
4. Type password (if you have one)	`Password:`	You won't see the password typed on the screen. In this example, `password1` is typed.
5. Press Return		Enters password.

If You See `TERM = (hp)`

Get your terminal type from your system administrator (terminal type tells the system how to interact with display terminals). For example, Series 300 high-resolution monitors need:

```
TERM = (hp) 300h Return
```

If you can't get the terminal type now, just press Return and continue. You may experience some display difficulties. The next time you login, be sure to use the correct terminal type.

Signing on to the System (Continued)

Problems You Might Encounter

Table 2–3 shows some problems that you might see during the login procedure.

Table 2–3. Problems During Login

If you see the message ...	It means ...
`Login incorrect.` `login:`	A typing mistake was made while typing your user name or password; try again. If you keep having problems, see your system administrator.

`Your password has expired. Choose a new one.`
`Changing password for` *your_user_name*
`New Password:`
	Type a new password—see the next module for how to do this.

`Maximum number of users already logged in.`	You'll have to wait until someone else logs out before you can log in.

For More Information

See *A Beginner's Guide to HP-UX* for more detail on logging in.

Creating/Changing Your Password

After you login, you should see a prompt similar to:

```
$
```

If your system is configured to automatically start X Windows, you can follow the instructions in this module without knowing much about X Windows. If you have trouble, see *Using the X Window System* manual.

After login, you are in HP-UX. This won't pose a problem if you are unfamiliar with HP-UX, you only need to do two things:

- Change your password
- Start BASIC/UX.

Your first task is to create a password. Change this password periodically to protect your work.

Choosing a Password

Choose a password with the following constraints:

- Must contain at least six characters
- At least two characters must be letters (upper- or lower-case)
- At least one character must be numeric or a special character (for example, @, -, _, or $)

Examples

Here are some sample passwords using the above constraints. Create your own password such that you remember it, but it is not easy for others to guess.

- `number1`
- `$money$`
- `r@@t@t@`
- `super-man`
- `24^gold`

Creating/Changing Your Password (Continued)

Changing Your Password

Follow the steps in Table 2–4.

Table 2–4. Creating or Changing Your Password

When you see this ...	Type this ...	Comments
`$`	`passwd` Return	Make sure `passwd` is lower-case (the `$` is a system prompt; your's might be different).
`Changing password for leslie` `Old password:`	Type old password if you have one.	If you are creating a password, you won't see this prompt. Also, you see your user name instead of `leslie`.
`New password:`	Type your new password.	Press Return when finished.
`Re-enter your password:`	Type the same password again.	The system is making sure you remember your password.

The next time you login, you must use your new password.

Problems You Might Encounter with passwd

The table below shows some possible errors you might see.

Table 2-5. Password Errors

Error Message/Problem	Do this ...
If you get stuck in `passwd` and you want to exit without changing anything ...	Press Break.
`Sorry.`	Typed your "old" password incorrectly. Start the procedure again.
`Password is too short - must contain at least 6 digits`	You typed a password less than 6 characters long.
`Password must contain at least two alphabetic characters and at least one numeric or special character.`	Your password doesn't conform to the rules described earlier.
`They don't match; try again.`	The second time you typed your password, you typed it different than the first.
`Too many failures - try later.`	You made more than 2 mistakes when typing your password. Start the procedure again.
`Password must differ by at least 3 positions`	Make sure your new password is different from your old password by at least three characters/numbers.

For More Information

For more details on `passwd`, see the *HP-UX Reference*.

Starting BASIC/UX

Your system administrator can set up the system to have BASIC/UX automatically loaded. If you see a screen similar to the one shown in "What the Screen Looks Like," BASIC/UX is already running.

Start BASIC/UX by typing:

```
rmb Return
```

If X Windows is already running, you'll see a new window appear; see Chapter 3 for more details.

What the Screen Looks Like

The following screen (or window, if X Windows is running) shows the BASIC/UX startup screen:

```
-------------------------------------------------
|               HP BASIC/UX 5.5                 |
| Copyright Hewlett-Packard Company 1981, 1982, |
|           1983, 1984, 1985, 1987, 1988        |
-------------------------------------------------
|                  HFS 5.1                      |
| Copyright Hewlett-Packard Co. 1987, 1988,     |
| Copyright The Regents of the University       |
|           of California 1979, 1980, 1983      |
| Copyright AT&T 1980, 1984                     |
-------------------------------------------------
|            RESTRICTED RIGHT LEGEND            |
| Use, duplication, or disclosure by the U.S.   |
| Government is subject to restrictions as set  |
| forth in subdivision (b)(3)(ii) of the Rights |
| in Technical Data and Computer Software       |
| clause at 52.227-7013.                        |
|              Hewlett-Packard Company          |
|   3000 Hanover Street, Palo Alto, CA  94304   |
-------------------------------------------------


HP BASIC/UX Main 5.5
```

Problems You Might Encounter

Possible problems:

- If you do not see the BASIC/UX screen, see your system administrator.
- If you are in X Windows and you see the BASIC/UX window with the message:

```
rmb: fatal internal error
```

  you may have too many processes for X Windows to start. Stop some processes or destroy some windows and retry BASIC/UX. If you still can't start BASIC/UX, see your system administrator.

For More Information

For details on the `rmb` command, see Appendix B. There are several options for this command that you might investigate once you become familiar with the system.

Leaving BASIC/UX (logout)

If you logged-in to the system, you must logout. Otherwise, you leave the system open for anyone to use.

Quitting BASIC/UX: QUIT

To exit BASIC/UX, type:

```
QUIT Return
```

If you are in X Windows, first make sure you quit BASIC/UX, then type Shift-CTRL-Reset to exit X Windows.

Logging Out of HP-UX: exit

If you see `login:` after leaving BASIC/UX, skip this section. Your system administrator set up the system to logout directly from BASIC/UX.

Otherwise, your system is in HP-UX and you need to logout from HP-UX as well. Type (note the commands are in lower-case):

```
exit Return
```

You are logged out, and you should see the:

```
login:
```

prompt again.

Problems You Might Encounter

Table 2-6. Problems with Logging-Out

If this happens ...	Do this ...
You don't see `login`:	Try typing `exit` Return again. See your system administrator if you have further difficulties.
You try the Workstation BASIC `SYSBOOT` command.	You receive an error message. You can only leave BASIC/UX with `QUIT`.
You try CTRL-d to leave BASIC/UX.	You receive an error message. You can only leave BASIC/UX with `QUIT`.

For More Information

See the *BASIC Language Reference* for more information on `QUIT`. See the *HP-UX Reference* for more information on `exit`.

Notes

3

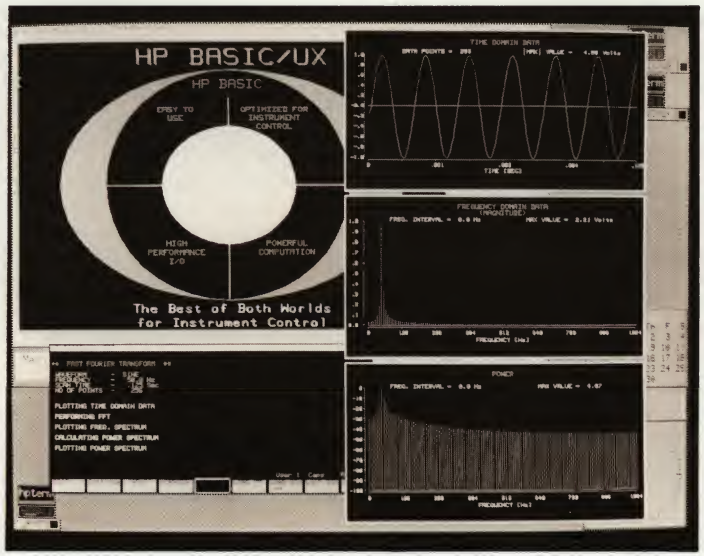
Using BASIC/UX Window Commands

X Windows is an optional product for BASIC/UX. If you are not running X Windows on your computer, you do not need to read this chapter.

This chapter does *not* describe how to use X Windows; see *Using the X Window System, Version 11.* It does, however, give you a summary of the common X Windows tasks, and describes BASIC/UX window commands.

A window system lets you

- Graphically organize your computing tasks
- Run several tasks at the same time
- Monitor more than one task on the display at the same time.



BASIC/UX in X Windows

Are You Already Running X Windows?

If you aren't sure what environment is running on your system, enter BASIC/UX (`rmb` Return) and type:

```
SYSTEM$ ("WINDOW SYSTEM") Return
```

Table 3–1. Determining Your Window State

If you see this ...	It means ...
`X Windows`	You are already in X Windows.
`Console`	You are not running X Windows.
`Terminal`	You are at a terminal and *cannot* run X Windows. (If you see a window system, it could be HP Windows/9000. Leave HP Windows/9000 with `wmstop` Return.)

Starting and Stopping X Windows

If X Windows is already running, skip ahead to "Running BASIC/UX in X Windows."

Prerequisites

Before you start X Windows, make sure of the following:

- You have a bit-mapped graphics display monitor.
- Your Mouse (or X locator device) is correctly plugged in.
- X Windows is not already running.
- HP Windows/9000 is not running.

Starting X Windows

To start X Windows:

1. If you are currently in BASIC/UX, type: `QUIT` Return
2. When you see the HP-UX prompt ($), type (make sure it's lower-case): `x11start` Return

Figure 3-1. What the Default X Window Display Might Look Like

Starting and Stopping X Windows (Continued)

HP-UX versus BASIC/UX Windows

When you start X Windows, a window is created (see Figure 3–1). The window can be thought of as a terminal in which you can execute HP-UX commands. When you execute the `rmb` command (see below), a BASIC/UX window is created. We'll call the window from which you executed `rmb`, the **root** window.

Running BASIC/UX in X Windows

1. Make sure the pointer is in an HP-UX window (called the root window).
2. Type:

   `rmb` [Return]

   to create a BASIC/UX window.

A BASIC/UX window is created (it may overlap the root window).

Stopping X Windows

If you are running BASIC/UX, exit with `QUIT`.

1. Stop all processes (all processes are killed when you exit X Windows, so you should terminate the processes normally first).
2. Press the [Shift] key and hold it down, press [CTRL] and hold it down, and then press [Reset]. For example:

   [Shift]-[CTRL]-[Reset]

Problems You Might Encounter

Table 3–2. Problems With Starting X Windows

If you see ...	It means ...
`sh: X11START: not found.`	Type `x11start` in lower-case.
`sh: xllstart: not found.`	Use 11 (numbers) not ll (letters).
`An X server is already running!`	You already have X Windows running.
`not enough shared memory`	See your system administrator for help.

For More Information

See *Using the X Window System, Version 11,* "How to Use the X Window System" chapter.

Summary of X Window Operations

The following table shows default X Window operations in the `uwm` window manager. For details, or if your window operations differ, see *Using the X Windows System.*

Table 3-3. Some X Window Operations

Operation	How to do it
Start X Windows	In HP-UX, type: `x11start` Return
Stopping X Windows	Shift-CTRL-Reset
Type in a window	Move the pointer with the Mouse until the pointer is in the window, then type.
Bring up the Window Manager menu	■ Move the pointer to a clear area (outside of a window) and press the *right* mouse button. ■ Press the Extend char key and the *right* Mouse button.
Selecting an item in the Window Manager menu	1. Bring up the Window Manager menu (hold down *right* button) 2. Move pointer down menu to item. 3. Release Mouse button.
Creating an HP-UX window	Select `Create Window` in the Window Manager menu.
Destroy an HP-UX window	With pointer in the window, type `exit` or CTRL-d

Table 3–3. Some X Window Operations
Continued

Operation	How to do it
Move a window	■ With pointer in window, Extend char and the *right* Mouse button. ■ Select `Move` in the Window Manager menu.
Resize a window	Select `Size` in the Window Manager menu.
Making a window an icon	With pointer in window, Extend char and *both* Mouse buttons.
Returning an icon to normal	With pointer in icon, Extend char and *both* Mouse buttons.
Bringing a window to the top of a stack	With pointer in window, Extend char and the *left* Mouse button.

Problems You Might Encounter

- Do not destroy the window from which BASIC/UX was started.
- If you somehow destroy all windows, select `Create Window` from the Window Manager menu and run `rmb` in the new window, or exit X Windows (with Shift-CTRL-Reset) and start X Windows again (`x11start`).
- If you are typing and you do not see anything being typed in a window, make sure the pointer is inside the window border.

For More Information

See the *Using the X Windows System* book for more details.

Creating and Destroying a Window

A window system lets you create several windows and run several tasks at the same time (one in each window). To send output to a created window, see "Redirecting Program Output to Other Windows."

- Windows created from the X Window environment (system menu for example) are HP-UX windows. These windows let you execute commands.
- Windows created from within the BASIC/UX environment are used as output devices within HP BASIC (you can send alpha or graphic output to these windows).

Creating BASIC/UX Windows

Move the pointer to the root window or another HP-UX window created with the X Window system menu, and type:

```
rmb Return
```

A BASIC/UX window is created. One HP-UX window is required for each BASIC/UX process; each BASIC/UX window represents a separate `rmb` process.

The window number for the first created BASIC/UX window is 600 (see "Listing Window Numbers" to list window numbers). It can be interchanged with `CRT` in many statements. For example, `PRINTER IS CRT` is equavilent to `PRINTER IS 600` (however, `MOVE WINDOW` and `CLEAR WINDOW` do not accept `CRT`).

Type: `QUIT` Return to destroy a BASIC/UX window.

Create a Window for BASIC/UX Text or Graphics Output

The `CREATE WINDOW` command lets you create windows for output (from programs, for example). The window created cannot be used to execute commands—only for output.

Example CREATE WINDOW Statements

For details on the syntax of CREATE WINDOW, see the *Language Reference*.

```
CREATE WINDOW 603,0,0,40,40;LABEL "SMALL"
```

window number
upper left corner
size (40x40 pixels)
label section in icon

```
CREATE WINDOW 645,450,100,500,400 ;RETAIN
```

window number
location is 450 pixels from left and 100 pixels from top
size (500x400 pixels)
saves a graphic image in a window

```
CREATE WINDOW 657,0,300,300,300, 15
```

window number
location is 300 pixels from top
size (300x300 pixels)
allows you 15 lines up or down in memory

Listing Window Numbers

To find out currently defined windows and their attributes, type:

```
LIST WINDOW [Return]
```

Each window is listed with its window number, X and Y positions, width, height, buffer size, retain graphics, whether or not it's an icon, and its label (for more information, see the *BASIC Language Reference*). Here is an example list:

```
WINDOW SYSTEM:  X
WINDOW    X      Y                          BUFFER        OPEN/
NUMBER   POS    POS   WIDTH   HEIGHT   SIZE    RET   ICON   LABEL
======   ===    ===   =====   ======   ======  ===   ====   ==========
600      100    50     640     338      28     No    Open   HP BASIC/UX: 600
601      200    350    640     400      0      Yes   Open   One: 601
```

Creating and Destroying a Window (Continued)

Destroying a Window

To delete a window created by BASIC/UX and all processes in that window:

- To delete a window created with `CREATE WINDOW`:

  `DESTROY WINDOW` *window_number* Return

  For example: `DESTROY WINDOW 603`
- `SCRATCH WINDOW` destroys *all* windows you defined from a given BASIC/UX process. The `rmb` window itself is *not* destroyed.
- To destroy a BASIC/UX window, type: `QUIT` Return.

Clearing a Window

This example shows how to clear the contents of window number 603:

```
CLEAR WINDOW 603
```

Problems You Might Encounter

See your display user manual for the maximum number of pixels on your display.

`Syntax error at cursor` means you typed something wrong in a window command. Check your syntax for possible typing errors.

For More Information

See *Using the X Window System* or the *BASIC Language Reference*.

Moving a Window

If you use windows for program output, you might wish to control the window location.

Prerequisites

You must have a window on the display created with `CREATE WINDOW`.

Moving Windows with MOVE WINDOW

```
MOVE WINDOW window_number,x,y
```

moves the window (with window number *window_number*) where the upper left corner is x pixels from the left side of the display and y pixels from the top of the display. For example,

```
MOVE WINDOW 601,100,400
```

moves window number 601 to 100 pixels from the left and 400 pixels from the top of the display.

For More Information

See *BASIC Language Reference*.

Redirecting Program Output to Other Windows

If you are running a program and wish to send the graphics output to another window, create a window and use the PLOTTER IS command.

Sending Graphics Output to Other Windows

Create a window using the CREATE WINDOW command described in "Creating and Destroying a Window." The following example shows how to send graphics output to a window with window number 645:

```
PLOTTER IS 645,"WINDOW"
```

When you run your program, the output of all graphics commands goes to window number 643. To return the output to the current BASIC/UX window, type:

```
PLOTTER IS CRT,"INTERNAL" or PLOTTER IS 600,"WINDOW"
```

Sending Text Output to Other Windows

For commands which list text output (such as CAT, LIST, etc.) you can send the output to another window. For example, you may wish to send a CAT list to another window so you can view it while doing something else in the first window. To send text output to window 604, for example:

```
PRINTER IS 604
```

When you type CAT, the output goes to window 604. To return the output to the BASIC/UX window, you must type:

```
PRINTER IS CRT
```

Problems You Might Encounter

Sending output to a window created smaller than the output truncates the output. To see the full output, re-size the window.

For More Information

See the *BASIC Language Reference* for information on PLOTTER IS.

Customizing X Windows

You can change the default colors, location, border width, and buffer size on BASIC/UX windows by adding parameters to an environment file called `.Xdefaults`.

Prerequisites

You must be familiar with an HP-UX editor (such as `vi`) to perform this task.

Edit the X Window Environment File: .Xdefaults

When you start X Windows, an `.Xdefaults` file is created for you. You can modify this system-generated file and add BASIC/UX-specific parameters.

The following steps show you how to create the default X Window environment file.

1. If you haven't done so, start X Windows to create the `.Xdefaults` file (at an HP-UX prompt: `x11start` Return). Then exit X Windows with CTRL-Shift-Reset.
2. Make sure you are in HP-UX. (If currently in BASIC/UX, type `QUIT` Return. If this logs you off the system, see your system administrator.)
3. Make sure you are in your home directory (`cd` Return takes you to your home directory).
4. Edit the file called `.Xdefaults`. For example, with the `vi` editor:

```
vi .Xdefaults Return
```

5. Add the following text to the file (in `vi`, type `i`):

```
rmb*Font:              hp8.10x20b
rmb*Foreground:        white
rmb*Background:        black
rmb*Geometry:          =80x25+50+100
rmb*BorderWidth:       2
rmb*Buffersize:        50
rmb*RetainedWindow:    0
rmb*WindowName:        HP BASIC/UX
rmb*Cursormode:        underscore
rmb*Resize:            0
```

6. Save the file (for example in `vi`, ESC :wq Return).

Changing Colors

Instead of the default `white` and `black` entries, you can add colors if you have a color monitor. You can find the possible color values with:

```
more /usr/lib/rgb.txt Return
```

The colors are listed in the right column. Use the colors without spaces. For example, use `DarkSlateGrey` instead of `dark slate grey` (note how capital letters are used).

To change color values for your entire display, see *Using the X Window System* for more details.

Problems You Might Encounter

If, when you start X Windows, your colors are not changed, check the file again to make sure the color is not misspelled or you do not have too many different colors that your display can support (ask your system administrator to look in the display's documentation for how many colors it can support). Also make sure you have r (read) and w (write) permissions on the file (see `chmod` in the *HP-UX Reference* for details).

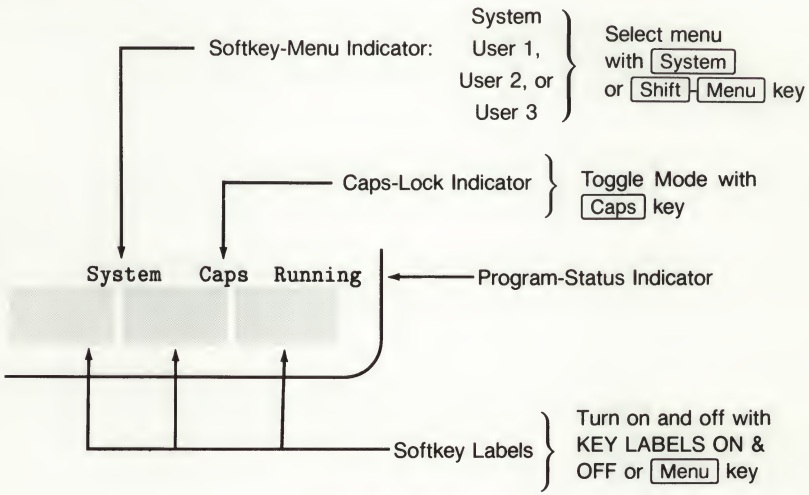
For More Information

For details on how to customize X Windows, see *Using the X Window System,* "User Level Customization".

4

Reading the BASIC/UX Screen and Executing Commands

BASIC/UX has the same environment as BASIC Workstation but with many enhancements.



Reading the Screen

Determining the System's Status

The BASIC screen displays information about the state of the system. When you run programs or walk up to a system with a program running, you may wish to determine its status.

Identifying the Status Indicators

Figure 4–1 shows the **status indicator** located at the lower right of your screen (or window).

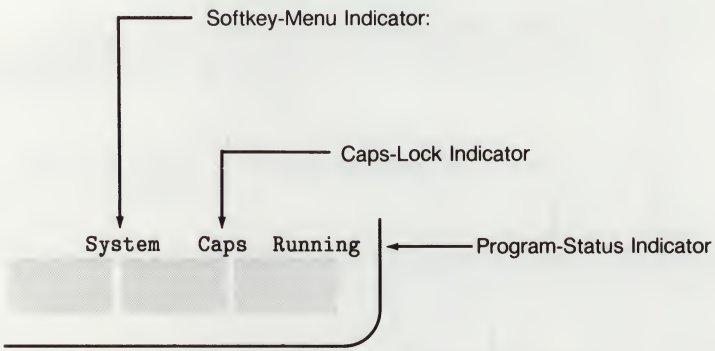


Figure 4–1. System Status Indicators

Reading the Program-Status Indicator

The Program-Status Indicator (Figure 4–1) tells you the status of any programs that may currently be running.

Table 4–1. Program-Status Indications

Status	Indicator	System State
`Idle`	(blank)	Program stopped; can execute commands.
`Running`	`R`	Program running; can execute commands.
`Paused`	`_`	Program paused; can execute commands; CONTINUE allowed.
`Transfer`	`I`	Program paused, but an overlapped TRANSFER (I/O) operation is still in progress; can execute commands.
`Input?`	`?`	BASIC program waiting for input from keyboard; cannot execute commands.
`Command`	`*`	System executing BASIC command; can enter one more command, but it is not executed until the current command is completed.
`Execute`	`E`	A command is being executed in the HP-UX environment.

Executing Commands

Commands tell the computer to perform a specific task. To execute a BASIC command, type the command and press [Return].

What is a Command?

Let's compare a BASIC command to a command for a person. There are two parts to a command.

- keywords (represent a pre-defined action)
- parameters (specific information for a particular command).

A **keyword** for a person might be "run". If you were commanded to "run", you would run until told to stop. If we added a **parameter** like "5 minutes", you would run for 5 minutes.

A keyword tells the computer *what* to do; a parameter tells *how* to do it. Sometimes parameters are omitted; in such cases, the computer assumes a parameter value (called the **default**).

Letter-Case Matters

You cannot mix letter-case when typing commands.

Table 4–2. How Letter-Case is Used

Correct Use of Letter-Case	Incorrect Use of Letter-Case
`CAT` or `cat`	`CaT`
`SYSTEM$("AVAILABLE MEMORY")`	`SYSTEM$("available memory")`
`EXECUTE "ls"`	`EXECUTE "LS"`

Note that `ls` in the above table is an HP-UX command. Letter-case is important when using HP-UX commands.

BASIC types in all upper-case by default. When you need to type lower-case, press [Shift] for individual characters or [Caps] to switch the letter-case.

Example Commands

Try the commands in this section.

- Note the status indicator as you execute commands.
- When you use the EXECUTE command in X Windows, the results are displayed in the root window which may be underneath the BASIC/UX window. Bring it to the top by moving the pointer to a clear area and pressing the *left* button.

Table 4-3. Example BASIC Commands

Type This Command	What It Does	Example Results
DATE$(TIMEDATE) Return	show BASIC date	26 Jun 1988
TIME$(TIMEDATE) Return	show BASIC time	09:53:53
EXECUTE "who" Return	display user names	julian ttyp0 Feb 29 08:57 fred tty02 Feb 29 08:32
KEY LABELS OFF Return	turn off softkey labels	(look at the bottom of your screen or window; KEY LABELS ON or Menu restores the labels)
SYSTEM$ ("VERSION:OS")	determine HP-UX version; A means single-user, B means multi-user	6.0B HP-UX

Executing Commands (Continued)

Recalling Commands

To repeat previous commands without typing the command again, press the f8 key (labeled `Recall` in the softkey label). See Chapter 8 for details.

Problems You Might Encounter

See Appendix E, "Error Messages," in the *BASIC Language Reference* for a list of errors. Here are some common ones:

Table 4-4. Possible Errors

Error Message	Typical Cause of Error
`Error 910 Ident not found in context`	Mixed letter-case, or mistyped a parameter.
`Error 949 Syntax error`	Mistyped command (check spelling).
`rmb-execute: WHO: not found`	If you use the `EXECUTE` command, be sure to use the proper letter-case within the quote marks. In this example, `WHO` should be `who`.
nothing on display	If you used the `EXECUTE` command in X Windows, all output goes to the HP-UX window from which the BASIC/UX window was started. Shuffle the HP-UX window to the top (see *Using the X Window System*).

For More Information

For information on keywords available, see the *BASIC Language Reference*. You might read about the commands before trying them.

Performing Calculations

In addition to using commands, you can use your system like a calculator.

Try Some Example Calculations

To perform arithmetic operations, use the operators in Table 4–5 (priority for computations are given in order of the table from lowest to highest).

Table 4–5. Arithmetic Operators for Keyboard Calculations

Operator	Operation	Example	Results
-	subtraction	2-4 Return	-2
+	addition	5.23+2.8-2 Return	6.03
/	division	5+3/2-1 Return	5.5
*	multiplication	3*3-1 Return	8
^	exponentiation	3^2*2-2 Return	16
SIN, COS, etc.	functions	SQRT(25)/5 Return	1
(...)	grouping	SQRT(125/5)+(2*3)/4 Return	6.5

Example Calculation Breakdown

Here is an example that shows priority for arithmetic calculations. We'll use the same example that was shown for the grouping example in Table 4–5.

`SQRT(125/5)+(2*3)/4`	*Example as you type it in.*
`SQRT(25)+6/4`	*Calculate the parenthetical groupings first.*
`5+6/4`	*Perform the SQRT function.*
`5+1.5`	*Calculate the division.*
`6.5`	*Perform the addition.*

Problems You Might Encounter

When you perform calculations that cause arithmetic errors, such as dividing by zero, the system responds with an error message.

Table 4–6. Possible Calculation Errors

Example	Error Response
`14/0` Return	`Error 31 Division by 0 or X MOD 0`
`SQRT(-3)` Return	`Error 30 SQRT of negative number`

For More Information

For more on performing calculations, see *BASIC Programming Techniques, Vol. 1: General Topics,* "Evaluating Scalar Expressions."

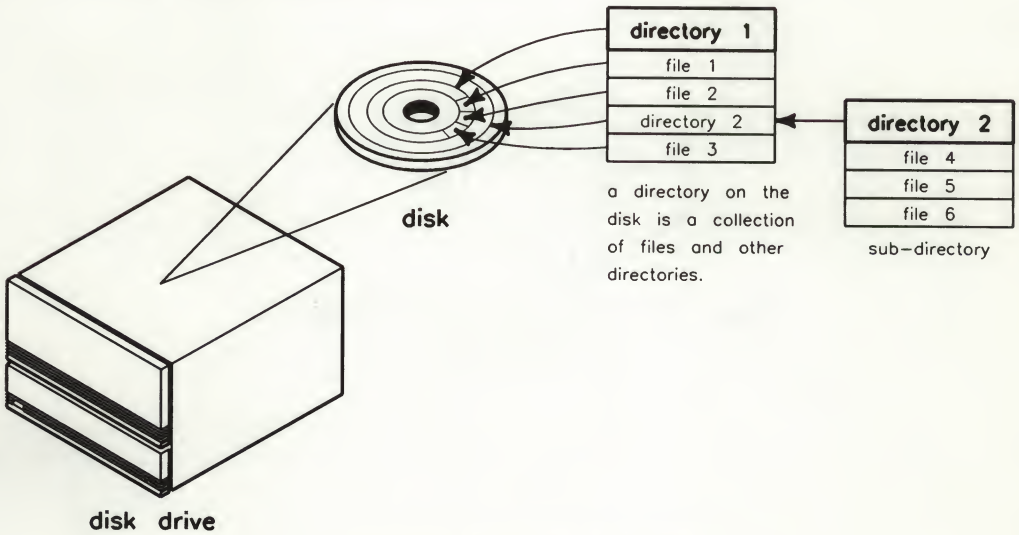
5

Working with Files, Directories, and Volumes

Mass storage devices (hard disks, flexible disks, cartridge tapes) store information such as programs and text. The information is organized in logical parts:

- **Files** contain programs or data. Each file has a name (unique within a directory) that allows you to access the information in the file.
- **Directories** contain files; directories also have unique names (HFS and SRM volumes only).
- **Volumes** contain directories.

The disk on which BASIC/UX is installed, is HFS-formatted (Hierarchical File System). This chapter describes how to work with files and directories on an HFS system. For information on other formats, see Chapter 11.



Directories and Files on a Disk

What Files and Directories Are

Computer information is placed in logical containers called files. Files in turn are organized into containers called directories. By organizing your files into directories, you gain easy access to information.

What Are Files?

Files are containers of information. Think of a file as a box with a name containing a BASIC program or textual information.

HFS file names can be up to 255 characters in length for Long File Name systems (LFN) or up to 14 characters for Short File Name systems (SFN). However,

- Don't use control characters that might mess up your terminal.
- Avoid creating file names the same as system commands or file names.
- The CAT command (listing the files in a directory) truncates file names longer than 14 characters.

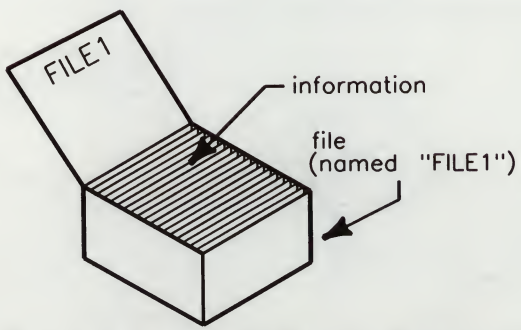


Figure 5–1. Files Are Named Containers of Information

What Are Directories?

A directory is a storage place for files, like a chamber or room might be a place to store boxes.

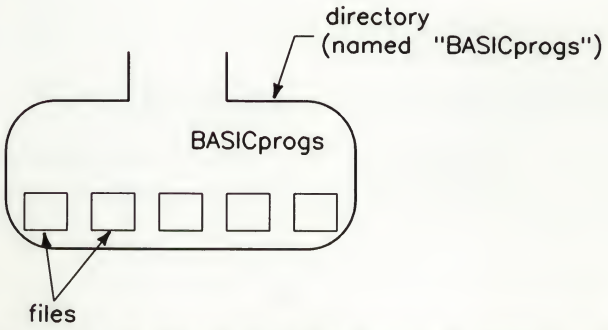


Figure 5-2. Directories Are Named Containers of Files

Some Misconceptions about Files and Directories

Comparing files to a box of information is an incomplete analogy if you use boxes as a storage container for everything. Files should contain specific information such as one program. Files also have names so you can access them, and files in a particular directory cannot share the same name.

A single directory should not be used as a container for all files; see "How Directories are Organized."

For More Information

To read more about files and directories, see *A Beginner's Guide to HP-UX,* the UNIX™ book shipped with your system, or one of the many books in a book store that discusses UNIX™ concepts.

How Directories are Organized

Directories are organized in a hierarchical system (on the HP-UX disk on which your BASIC/UX system is installed). That is, you can have many directories organized in such a way that the structure implies some kind of order.

Compare the Directory Structure to a Pyramid

In the previous module, we compared files to boxes and directories to chambers. Hooking directories together in a hierarchical structure is similar to connecting chambers in a pyramid:

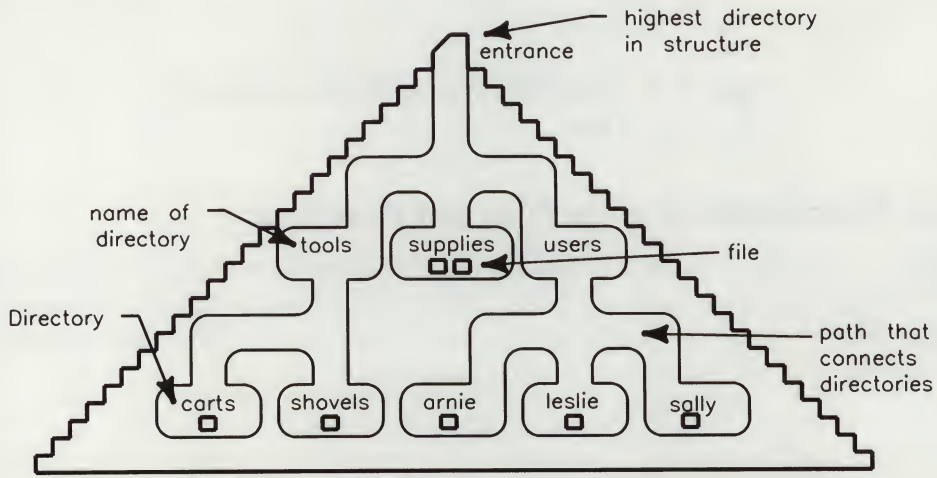


Figure 5-3. Directories Compared to a Hypothetical Pyramid

The ability to connect directories in this way allows you to organize your files. For example, the `leslie` directory at the bottom of the pyramid contains files that belong to Leslie. Table 5–1 explains further similarities between the directory structure and pyramids.

Table 5–1. Directories Compared to a Hypothetical Pyramid

Pyramid	Hierarchical Directories
Chambers contain boxes.	Directories contain files.
A chamber contains an entrance from above.	A directory has a **parent directory.**
A chamber may contain passageways to chambers below.	A directory may have subordinate **child directories.**
A pyramid has one entrance at the top.	The directory structure has a **root directory** at the "top".

How Directories are Organized (Continued)

Having a good idea of how to move within a pyramid (going from one chamber down to the next through a passageway) helps you understand how you can "move" within a hierarchical directory structure.

Applying the Analogy to the Actual Structure

Figure 5–4 shows the directory structure that could be used to represent the chambers and boxes of the pyramid in Figure 5–3. (Ovals represent directories; boxes represent files.)

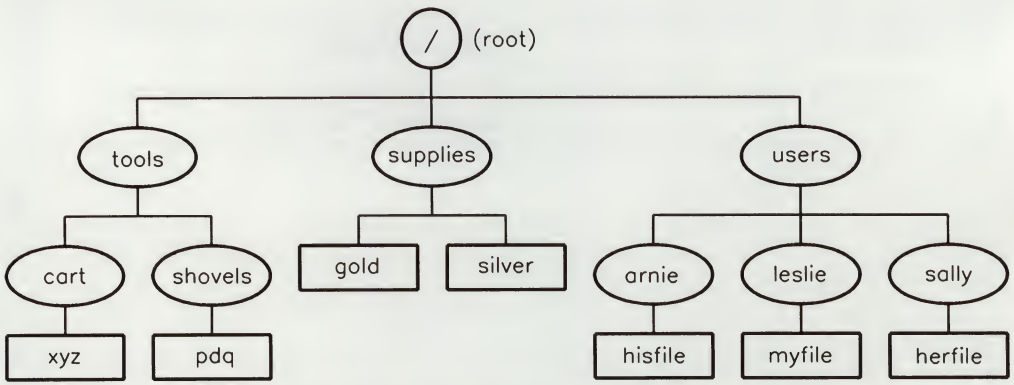


Figure 5–4. Directory Structure Mirroring the Pyramid Structure

When you begin using the system, you are assigned a default directory, called your **home** directory. You can access files in that directory by using only the files' name. To access files in other directories, you would have to indicate a **path** through the directory structure to the file (explained in "Referencing Files and Directories: Path Names").

Determining Your Place in the Structure

Try the following commands. They print the system's way of showing where you are in the structure. Press Return after you type a command.

Table 5-2. Determining Your Place in the Structure

Type this ...	What it does	Example Results
`EXECUTE "echo $HOME"`	Prints your HOME directory (displays in HP-UX window for X Windows)	`/users/arnie`
`SYSTEM$("MSI")`	Prints your current directory. All characters before `:HFS` (or `:REMOTE`) comprise the path name.	`/users/arnie/project1:HFS`

For More Information

A Beginner's Guide to HP-UX also uses the pyramid analogy.

Referencing Files and Directories: Path Names

Since files are located all over the directory structure, you need a way to reference them. **Path names** show the computer a way to get to a particular directory or file, like a map would show you how to travel through the pyramid to get to a particular chamber or box:

- **absolute path names** show the path to a directory or file starting from the root directory (the uppermost directory, symbolized by "/")
- **relative path names** show the path to a directory or file from the current directory.

Using Absolute Path Names

Figure 5–5 shows the absolute path name for each directory in its structure. Notice that "/" separates directory names, and the first "/" indicates the root directory.

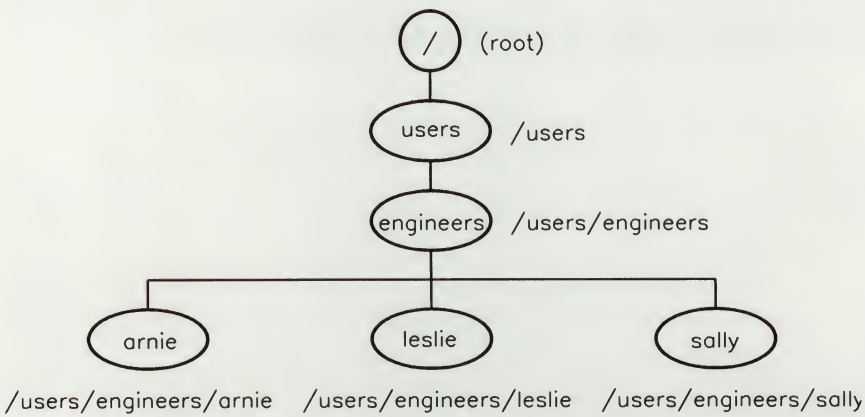


Figure 5–5. Absolute Path Names

Using Relative Path Names

Figure 5–6 shows relative path names from the directory `leslie` (`..` indicates the "parent" directory; `.` indicates the current directory).

Start at directory `leslie`. Moving up in the structure adds `..` to the path.

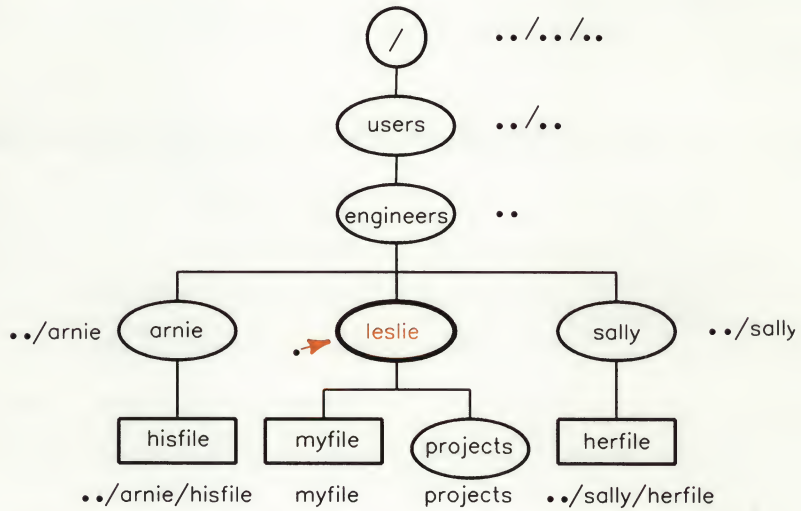


Figure 5–6. Relative Path Names from /users/engineers/leslie

For example, the path from `leslie` to `herfile` requires you to go up the structure to `engineers` (`..`), horizontally to `sally` (`../sally`), then down to `herfile` (`../sally/herfile`).

Referencing Files and Directories: Path Names (Continued)

Try Some Examples

Try these example commands. They show how to list the contents of directories from BASIC/UX.

Table 5-3. Example CAT Commands Using Absolute and Relative Paths

Type this ...	What it Does
`CAT` Return	List contents of current directory
`CAT ".."` Return	List contents of parent directory (relative)
`CAT "/users"` Return	List contents of `users` directory (absolute)
`EXECUTE "ls"` Return	Use the HP-UX `ls` command to list the current directory.

Example CAT Output

When you list the contents of a directory, you see a list similar to below.

```
/users/leslie:HFS
LABEL:
FORMAT: HFS
AVAILABLE SPACE:     166780
                  FILE     NUM    REC      MODIFIED
FILE NAME         TYPE     RECS   LEN DATE        TIME PERMISSION OWNER GROUP
=============== ===== ======= ===== ========= ===== ========== ===== =====
.profile        HP-UX      760     1 23-Jun-88 11:18  RWX------   204    10
.x11start       HP-UX     1729     1 23-Jun-88 11:05  RWX------   204    10
.Xdefaults      HP-UX      456     1  8-Jul-88  7:29  RWX------   204    10
reports         DIR          7    32 19-Jul-88 13:13  RWXRWXRWX   204    10
backup          DIR         39    32 27-Jun-88 13:41  RWXRWXRWX   204    10
notes           HP-UX      106     1 14-Jul-88 16:28  RW-RW-RW-   204    10
SPL_PROGRAM     PROG         1   256 14-Jul-88 16:28  RW-RW-RW-   204    10
ASCIIPROG       ASCII        1   256 14-Jul-88 16:28  RW-RW-RW-   204    10
```

Explaining CAT Output

Table 5-4. Meaning of CAT Entries

Column Heading	Examples	Description
FILE NAME	`notes, ASCIIPROG`	Names of files in the current directory.
FILE TYPE	`PROG, ASCII, HP-UX, DIR`	Type of file: ■ PROG is a BASIC program. ■ ASCII is an ASCII file. ■ HP-UX is an HP-UX type file. ■ DIR is a directory (sub-directory to current).
NUM RECS	`1, 760`	Number of records.
REC LEN	`256`	Record length.
MODIFIED	`14-JUL-88`	Date (and time) file was last modified/created.
PERMISSION	`RW-RW-RW-`	Who is allowed to R(ead), W(rite), or eX(ecute) the file for the file's Owner, Group, or Other (everyone on the system). Execute permission is also referred to as SEARCH permission.
OWNER	`204`	Numerical ID of the file's owner.
GROUP	`10`	Numerical ID of the file's group.

Problems You Might Encounter

A CAT of an HFS directory requires R (read) and X (search) permissions on the directory, as well as X (search) permissions on all parent directories. See "Permitting HFS File Access" for information on permissions if you receive an error when executing `CAT` (for example, `ERROR 183 Permission denied`).

For More Information

See the *BASIC Language Reference* for more detail on `CAT`. See *A Beginner's Guide to HP-UX* for more on path names.

Creating Directories

To organize your own files, you can create additional directories from your home directory with the `CREATE DIR` command.

Prerequisites

Generally, you should only create directories subordinate (below) your home directory unless you are the system administrator.

Creating Directories from Your Home Directory

Try the examples from your home directory—we'll use the directories in the next module for learning how to move from directory to directory.

- `CREATE DIR "SAMPLE"` Return
- `CREATE DIR "TRAIL"` Return
- `CREATE DIR "SAMPLE/PROJECT1"` Return
- `CREATE DIR "SAMPLE/PROJECT1/FILES"` Return

When you complete these commands, you create a directory structure shown in Figure 5–7.

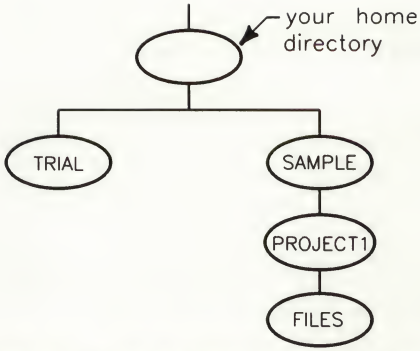


Figure 5–7. An Example Directory Structure You Create

Creating Directories with Absolute Path Names

If you are the system administrator, you may wish to create directories outside your home directory. If you use absolute path names to create a directory, only the last directory name in the path can be the new directory. For example, if you type:

```
CREATE DIR "/tmp/sample"
```

`/tmp` *must* already exist (you'll see `ERROR 56 File name is undefined` if it doesn't exist).

Problems You Might Encounter

You must have the correct permissions for each level of directories in the path down to the parent of the directory you are creating. See "Permitting HFS File Access" for more information on permissions.

For More Information

See the *BASIC Language Reference* for more on the `CREATE DIR` command.

Changing Directories

Making another directory your current directory is done with the MSI statement (stands for "MASS STORAGE IS").

Prerequisites

Before trying the examples in this module, find the path name for your current directory so you can return to it. Type:

```
SYSTEM$("MSI")
```

For example, /users/leslie:HFS. The path name is: /users/leslie

Changing Directories with Relative Path Names

If you have to move to a directory in "close proximity" to your current directory, you can use a relative path name. Table 5–5 lists some commands can try from your current directory based on the example from "Creating Directories with Absolute Path Names."

After each command, type: CAT and check the path name at the top of the list. Then type: CLS to clear the screen (makes it easier to see the next list).

Table 5–5. Changing Directories with Relative Path Names

Example	Explanation
MSI "TRIAL"	Move "down" to TRIAL directory.
MSI "../SAMPLE"	Move "up" to home directory and then "down" to SAMPLE directory.
MSI "PROJECT1/FILES"	Move down two levels to FILES directory
MSI "../../.."	Move up three levels back to the directory from where you started.

Type the SYSTEM$("MSI") command again to make sure you are in your home directory. If you aren't, type the MSI command with your home directory path name. For example: MSI "/users/leslie"

Changing Directories with Absolute Path Names

Table 5–6 shows you some examples that use directories from Figure 5–6 (note that these examples won't necessarily work on your particular system).

Table 5–6. Changing Directories with Absolute Path Names

Example	Explanation
`MSI "/"`	Move directly to the root directory.
`MSI "/users/engineers"`	Move directly to `/users/engineers`.
`MSI "/users/engineers/arnie"`	Move directly to the `arnie` directory.

Changing Directories to LIF Disks

You can change your directory to a LIF disk by including the the mass storage volume specifier (*msvs;* see Chapter 11) of the LIF disk (see your system administrator for the disk's msvs). For example:

```
MSI ":,700"
```

where `:,700` is the msvs. To return to the HFS directory from LIF disk, type:

```
MSI ":HFS" Return
```

Problems You Might Encounter

`Permission denied` means you tried to get into a directory that has been protected. The "owner" of the directory must change the permissions before you can change to that directory.

To determine your home directory's path name: `EXECUTE "echo $HOME"`

For More Information

See the *BASIC Language Reference* for more information on the `MSI` keyword.

Permitting HFS File Access

The PERMIT command lets you define who can read or modify your HFS files.

What Are Access Permissions?

Access permissions tell you who can use a file or directory and what they can do to a file or directory. There are three categories of users on your system:

- OWNER: you are the owner of files you create or copy.
- GROUP: a group of users defined by the system administrator.
- OTHER: everyone on the system except yourself and your group.

Each user can access a file or directory depending on these three criteria:

- READ: able to look at or copy a file (or look in a directory).
- WRITE: able to change or delete the contents of a file or directory.
- SEARCH: able to include the directory in a path when accessing another file or directory. This is sometimes referred to as "execute" permission.

Using these permissions, you can define who you want to use your files or directories, and what they can to do to the files or directories.

Using the PERMIT Statement

You can use the PERMIT statement to assign and remove access permissions of a file or directory. Table 5–7 shows some example PERMIT statements; note that these commands must be typed on a single line and Return should be pressed only after typing the entire command.

Table 5–7. Protecting Files

Example	Explanation
`PERMIT "File"; OWNER:READ,WRITE`	Allow the owner (the person who created the file) to READ and WRITE (make changes to) the file "File". GROUP and OTHER permissions are unchanged.
`PERMIT "File"; OWNER:READ,WRITE; OTHER:READ`	Allow the owner to READ and WRITE "File", and allow all other users to only READ the file. GROUP permissions are not affected.
`PERMIT "File"; GROUP:READ`	Allow the people in the file's GROUP to READ "File"; does not change permissions for OWNER or OTHER.
`PERMIT "File"`	Restore READ and WRITE permissions for all users.
`PERMIT "Directory"`	Restore READ, WRITE, SEARCH permissions for all users for a directory. (You can protect directories in the same way as files.)
`PERMIT "Dir1"; OWNER:READ,WRITE,SEARCH`	Allow the owner to read from, write to, or search the directory `Dir1`. GROUP and OTHER permissions are unaffected.

For More Information

For details on the PERMIT statement, see the *BASIC Language Reference*. For information on SRM permissions, see *Using the BASIC System*. To lock files so they can be accessed by only the person who locked the file, see the *Programming Techniques*, "Data Storage and Retrieval" chapter, "Locking Files" section.

Copying Files

The COPY statement allows you to duplicate files. Any type of file may be copied.

Table 5-8. Copying Files and Volumes

Example	Explanation
`COPY "ExistFile" TO "NewFile"`	Duplicate `ExistFile` into `NewFile`
`COPY "F2" TO "/users/dan/F2"`	Copy the file `F2` to another directory

Concerns

To copy a file, you must have the correct permissions (described in "Permitting HFS File Access") that allow you to READ, WRITE, and SEARCH. You cannot copy whole directories, although you can copy a file from one directory to another. Unlike the HP-UX `cp` command, you cannot specify a directory name only to copy a file to that directory; you must include the file name.

If you are copying from an SRM or LIF disk an HFS volume, use the `COPY` command in BASIC and not the HP-UX utilities `srmcp` or `lifcp`. Using `srmcp` or `lifcp` causes incorrect behavior due to special headers on HFS files. See Chapter 11.

Problems You Might Encounter

If the new name (name of the copy) is the name of an existing file, you receive an error message: `ERROR 54 Duplicate file name`.

For More Information

See the *BASIC Language Reference* for more information on `COPY`.

Renaming Files

Renaming files allows you to change the name of a file. Make sure the new name is not the name of an existing file; you get an error message otherwise.

Table 5–9. Renaming a File

Example	Explanation
`RENAME "MyFile" TO "YourFile"`	Renames the file `MyFile` to a file named `YourFile` in the current directory.
`RENAME "This" TO "../engineers/That"`	Renames the file `This` to a file named `That` in the `engineers` directory: actually moves the file to the new directory.
`RENAME "MyFile" TO "../MyFile"`	Moves the file `MyFile` to the parent directory

Problems You Might Encounter

To rename an file, you must have the correct permissions (described in "Permitting HFS File Access") that allow you to READ, WRITE, and SEARCH in the directory in which you are renaming the file.

For More Information

See the *BASIC Language Reference* for more information on `RENAME`.

Purging (Deleting) Files or Directories

Purging a file deletes the directory entry for the file. Once a file is purged, there is no way of retrieving the information it contained.

Purging Files and Directories

The following statement removes the file "Old_stuff" from the current directory:

```
PURGE "Old_stuff"
```

Example Purge Statements

If you created directories in the "Creating Directories" module, try the following examples (it's assumed you are in your home directory):

Table 5–10. Examples of Purging Directories

Example	Description
`PURGE "SAMPLE"`	You get an error message because the directory is not empty: it contains other directories.
`PURGE "SAMPLE/PROJECT1/FILES"`	Purge the `FILES` directory.
`PURGE "SAMPLE/PROJECT1"`	Purge the `PROJECT1` directory.
`PURGE "SAMPLE"`	Now you can purge `SAMPLE`.
`PURGE "TRIAL"`	Purge the `TRIAL` directory.

Restrictions for Purging Files and Directories

The PURGE statement can be used for removing files and directories. Here are some restrictions on using PURGE to remove files:

- In order to use PURGE, you must have W (write) permission on the parent directory, and X (search) permission on all superior directories.
- You cannot purge the current directory.
- Directories must be empty before you purge them (cannot contain any files or other directories).

For More Information

See the *BASIC Language Reference* for more information on the `PURGE` command.

Linking Files

LINK lets you link a new file name on an HFS volume to an existing file on the same volume. This command saves disk space, and lets you reference one main file with several file names.

Prerequisites/Constraints

LINK can only be used with an HFS volume. You cannot link files from one disk to another disk (however, HP-UX does support symbolic links; see the entry for cp(1) in the *HP-UX Reference*).

Using LINK

The following example links the file `COREFILE` to another file name, `NEWFILE`:

```
LINK "COREFILE" TO "NEWFILE"
```

When you access `NEWFILE` (with an OUTPUT command, for example), the actual file accessed is `COREFILE`. So in essence, LINK is much like COPY, except the actual file is not copied, and disk space is saved.

Considerations When Using LINK

The following items are important to know when using LINK:

- If the file is BDAT, ASCII, or HP-UX, an OUTPUT to the file changes its contents.
- An ENTER command on any linked file reflects any changes to the core file.
- If you RE-STORE or RE-SAVE to a file linked to other files, a new file is created and the link to the original data is broken. The RE-STOREd or RE-SAVEd file is changed, but the original data and file names referring to the original data are not changed.
- Use LINK when you want to save disk space and when you want changes to be reflected in all linked files.
- Use COPY when you want to have two separate files, and you want changes to be reflected in only one copy of the file.

For More Information

See the *BASIC Language Reference* for more on LINK.

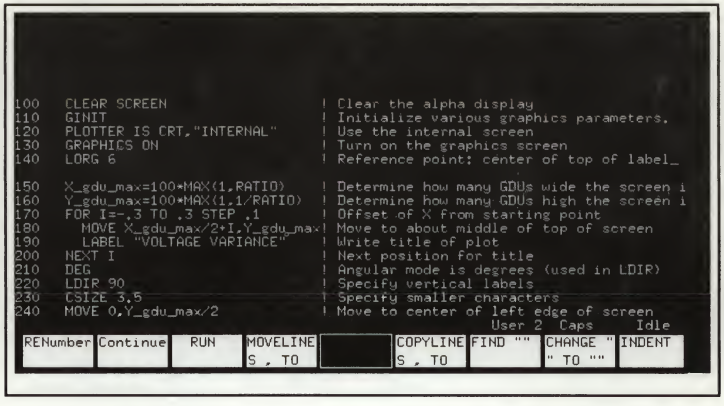
LINK is also programmable. See *Programming Techniques* for details.

Notes

6

Editing a Program and Storing It on a Disk

If you plan to only run application programs, you probably don't need to read this chapter. This chapter is for those who wish to write or modify BASIC programs.



Example BASIC Program in Edit Mode

Entering and Leaving the EDIT Mode

EDIT lets you write a BASIC program and checks your program syntax as you edit.

Entering EDIT Mode

To enter the EDIT mode, type:

```
EDIT [Return]
```

What the Screen Looks Like

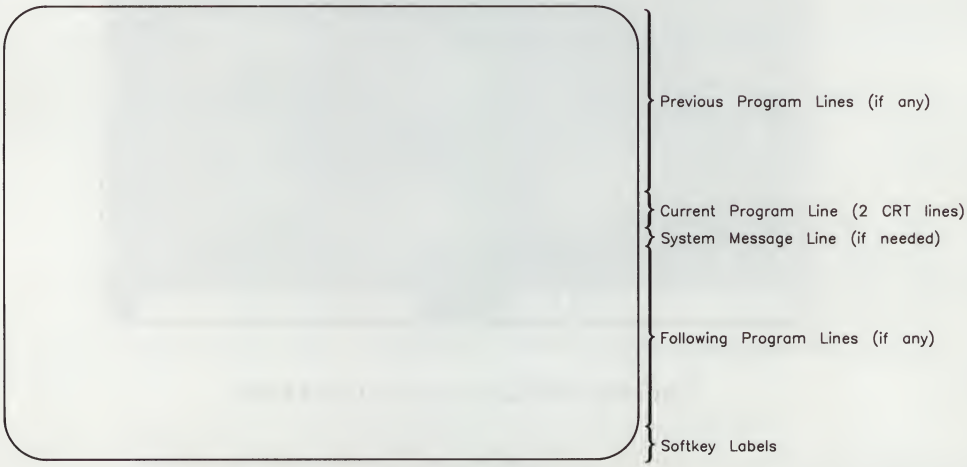


Figure 6–1. EDIT Mode in a BASIC/UX Window

Leaving EDIT Mode

Table 6–1 shows how you can leave EDIT mode.

Table 6-1. Leaving EDIT

Type or Press ...	Description
Stop or Clear display	Most common way to exit the EDIT mode
Some Softkeys	Pressing a softkey to start an operation (such as RUN) leaves the EDIT mode
Display operation	Leaves the EDIT mode to perform an operation that uses your display (such as LOAD, GET, LIST, CAT)

Concerns/Restrictions

EDIT is *not* a programmable statement. EDIT mode cannot be entered if a program is currently running (stop the program, then enter EDIT mode).

When you do not add parameters to the EDIT command:

- EDIT begins at line 10 if there is no program in memory or you just executed a LOAD command.
- If you just ran a program with errors, EDIT places the line that generated a run-time error as current line.
- When editing the same program several times, EDIT assumes the current line as the line last edited from the previous EDIT session.

For More Information

See the *BASIC Language Reference* for more details on EDIT.

Entering Program Lines with the Editor

Program lines are entered at the current line

- A blank line is directly below the current line.
- Locate the cursor by pressing the arrow keys or the space bar.

If There Is a Program In Memory

If the screen is full of text (a program), you have a program in memory. If you don't want to edit this program, you need to clear EDIT memory:

Shift-Clear line SCRATCH Return

Entering Program Lines by Typing

As you type, the characters appear on the current line. As you press Return at the end of each complete program line:

- The line is saved into EDIT memory (if you don't press Return, the line disappears when you move to another line)
- A new line with a new line number appears.
- A message appears if your line has an error.

How Syntax Errors are Handled

BASIC checks your program syntax as you type in lines. For example,

```
10  PRINT "COUNT FROM ONE TO TEN. Return
```

generates "**Error 985 Invalid quoted string**", and places the cursor under the quote ("). To correct the line, you would press Shift-➤ to move the cursor to the space following the last character (.) and type: " Return.

Examples

Type the following program. Note:

- If you type commands in lower-case, they are converted to upper-case once you press [Return] (except for characters within quote marks).
- When you press [Return] a new line number is generated automatically.
- To correct simple typing mistakes, use [Back space] to back over the text and re-type the line. If you don't see the mistake until after you press [Return], see "Correcting Typing Mistakes."
- If you wish to run the program when finished, press the RUN softkey ([f3]).

```
10 PRINT "Count from one to ten."
20 FOR I=1 to 10
30  PRINT I
40 NEXT I
50 PRINT "It works!"
60 END
70
```

Scrolling Through a Program

Editing a program line requires you to either:

- Type the line number and text in the command line (or in EDIT by first typing Shift-Clear line). It is automatically inserted in its proper order or it replaces an existing line number.
- Enter EDIT and edit the line.

To get to a particular program line, you must scroll to that line (you may also search for a program line by typing part of it and having BASIC find the line; see "Finding Textual Patterns").

Keys For Scrolling

Table 6–2 shows you which keys let you scroll through program text:

Table 6–2. Keys for Scrolling Program Text

Editing Key	Explanation
▼	Scroll the program down one line (cursor moves up one line).
▲	Scroll the program up one line (cursor moves down one line).
Shift-▲	Scroll the program up so the end of the program is displayed and the next available line number is shown on the current line.
Shift-▼	Scroll the program down so the beginning of the program is displayed and the first program line is the current line.
Mouse	In console display only (not in X Windows), use the mouse to scroll.

Examples

Try scrolling with the example program from the "Entering Lines in the Editor" module.

Problems You Might Encounter

If you make changes to a line and scroll off it without pressing Return, the changes are lost.

For More Information

For other "search" methods, see "Finding Textual Patterns."

Making Program Line Changes

To make changes in a program, scroll to the program line and use one of the features described in this module.

Prerequisites

Enter the EDIT mode and scroll to a line with the keys described in "Scrolling Through a Program."

Editing the Current Line

The following table describes the features used to edit the current line.

Table 6-3. Keys to Edit Current Line

Editing Feature	Explanation
Type at cursor	Overwrite any characters at the cursor
[➤], [◄], [Backspace]	Move cursor one character in indicated direction
[Shift]-[◄], [Shift]-[➤]	Moves cursor to the beginning or end of the line
Mouse	In the console mode only (not in X Windows) use the mouse to move the cursor left and right.
[Insert char]	Insert text at cursor (press [Insert char] again to end insert mode)
[Delete char]	Delete character at cursor
[Clear line]	Delete characters from cursor to end of current line
[Shift]-[Clear line]	Clears the current line (beginning to end)
[f5], [Shift]-[f5]	(in `System` menu) Set tabs or clear tabs (see softkey menu).
[Tab], [Shift]-[Tab]	Moves cursor to next or previous tab position

To edit a line without scrolling, press [Shift]-[Clear line] to clear a line, and then type the line number and the text as you would an inserted line. Press [Return], to insert the line in its proper order.

Examples

Use the sample program in "Entering Program Lines with the Editor" and make these changes:

1. Move to line 10 and change the text to read:

   ```
   PRINT "Print numbers from 1 to 10"
   ```

2. Move to line 50 and change the text to read:

   ```
   PRINT "Finished"
   ```

There are many ways to complete the above operations; find what you feel is the easiest for you. Here is what the file should look like:

```
10 PRINT "Print numbers from 1 to 10"
20 FOR I=1 to 10
30  PRINT I
40 NEXT I
50 PRINT "Finished"
60 END
70
```

Inserting, Deleting, and Recalling Program Lines

Use the features in this module to delete entire program lines and insert new ones.

Edit Program Lines

Table 6–4 shows you the keys to add a line between two existing program lines, delete a line, and recall a deleted line.

Table 6–4. Keys to Insert, Delete, or Recall a Program Line

Key	Explanation
Insert line	Open a new line before the current line and number it between the previous line and the current line. You are placed in the insert mode and can begin typing the new line. To end the insert mode, press Insert line again.
Delete line	Delete current line.
Recall	Restore last deleted line. To store it as part of the program, press Return.
Shift-Recall	Cycle through the most recently deleted, entered, or executed lines. You can press Shift-Recall until you see the desired line, and then press Return to store the line into the current program.

Examples

Use the example program in "Entering Program Lines with the Editor":

1. Insert before line 10: `PRINT "Begin."`

   Don't forget to press Return after you type the new line to save it.

2. Delete line 50 (`PRINT "Finished"`).

3. Recall the line you just deleted, and press Return to save it.

Storing a BASIC Program

After editing a program, you can store (save) the program on the disk.

STORE a Program from Within EDIT Mode

While still in the edit mode,

1. Press Shift-Clear line
2. Type: `STORE "`*Prog*`"` Return

where *Prog* is any name you give a program. For example, `STORE "SAMPLE"`

You can also use the `STORE` softkey (see "Using Softkeys" chapter).

Check the Program Was STOREd

Type:

`CAT` Return

The `CAT` command is discussed in Chapter 5. Look for the file name you just saved.

Storing to an Existing File

If you are editing a file that's already been stored, use `RE-STORE` instead of `STORE`. For example, `RE-STORE "SAMPLE"`

SAVE an HP-UX File

If you need to save the file as HP-UX file type (ASCII characters), use `SAVE` instead of `STORE` (used for data files, for example). See Chapter 9 and the *BASIC Language Reference* for details.

Finding Textual Patterns

Instead of scrolling to a program line, you can search for a textual pattern. FIND searches from the current line to the end of the program.

Prerequisite: Typing Commands in EDIT

To type commands in EDIT so you don't type into a program line:

Shift-Clear line

Only the current line is cleared to type; it does not destroy the line (Delete line does that).

Searching with FIND

Table 6-5. FIND Examples

Example	Explanation
`FIND "Works"`	Searches for the string "Works" and makes the first line containing the string the current line. Be sure you use the correct string—the correct letter-case.
`FIND "pattern" IN 200,650`	Searches for "pattern" on lines 200 through 650
CONTINUE	If there are several lines with the same pattern, CONTINUE leaves the line unchanged and searches for the next occurrence.

Leaving the FIND Mode

You remain in FIND mode until you

- Enter a line after changing its line number
- Press Break or any key that cancels EDIT mode.

For More Information

See the *BASIC Language Reference*.

Copying, Moving, Changing, and Deleting Blocks of Text

Making global changes to programs can help if you have more than one line to change. To type the following commands, press: Shift-Clearline first.

Table 6-6. Global Changes

Example	Explanation
`COPYLINES 180,220 TO 5205`	Copy lines 180 through 200 to a location beginning at line 5205 (lines 180-200 don't change)
`MOVELINES 32,127 TO 453`	Move lines 32 through 127 to a location beginning at line 453 (lines 32-127 don't exist after the move).
`CHANGE "Old text" TO "New text"`	Search for `Old text`, delete it and replace with `New text`
`CHANGE "old" TO "new" IN 1,250`	Search for `old` between lines 1 and 250 and replace the first occurrence with `new`
`CHANGE "old" TO "new" ; ALL`	Search for every occurrence of `old`, delete it and replace with `new`
`DEL 100,200`	Delete lines 100 through 200
`DEL Block2,327`	Delete all lines from the label `Block2` through line 327

Copying, Moving, Changing, and Deleting Blocks of Text (Continued)

Problems You Might Encounter

Table 6–7. Problems when Copying, Moving, Changing, or Deleting

If this happens ...	This might be the cause ...
`ERROR 40 Bad REN, COPYLINES, or MOVELINES`	You tried to copy or move to an existing line.
`"`*old*`" to "`*new*`"?`	When changing text, you are asked to confirm the change with `Y`.
Only the first occurrence of *old* is changed when changing a block of text	Add `;ALL` to the end of the CHANGE command changes all occurrences of the text
`ERROR 963 Command only`	These commands are not programmable.

For More Information

See the *BASIC Language Reference*.

Making Your Programs Easier to Read

Programs with consistent line number increments, indents, and comments are easy to read.

Placing Comments in the Program

Include your comments after the ! character. In the following example, the string `!End the program` is ignored when the program is run.

```
200 PRINT "Finished"  !End the program
```

Renumbering and Indenting Program Lines

Table 6–8 shows you commands to renumber and indent program lines.

Table 6–8. Commands to Make Programs Readable

Command	Explanation
`REN`	Renumber the program lines using 10 as the first line, and increment lines by 10 (default).
`REN 100,5`	Renumber the program using 100 as the first line, number and increment lines by 5.
`REN 100,5 IN 100,200`	Renumber only in the line range of 100 to 200; make the first line of this segment 100, and increment by 5.
`INDENT`	Indent lines to show nesting of branching constructs (FOR ... NEXT, REPEAT ... UNTIL, etc.).
`INDENT 7,2`	Place the first character of the outermost construct in column 7 and space each indentation by 2 characters (INDENT 6,2 is the default).
`INDENT 7,0`	Remove all indentation.

For More Information

For details on these commands and any limitations, see the *BASIC Language Reference*.

Entering Non-Alphanumeric Characters

Characters that don't appear on the keycaps (such as control or escape characters) can be typed by using [f7] (the `ANY CHAR` key in the SYSTEM menu).

Using the ANY CHAR Softkey

For example, let's assume your keyboard does not have the vertical bar character (|). Press [f7] (`Any Char`). The following message appears below the current line:

```
ENTER 3 DIGITS, 000 THRU 255
```

The decimal ASCII code for a vertical bar is 124. If you type 124, a vertical bar appears at the cursor position, and the message goes away.

Where To Find an ASCII Table

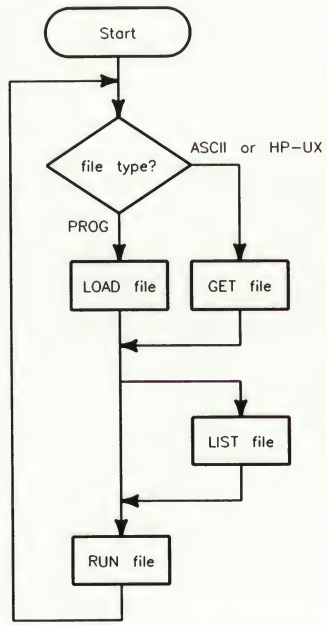
See Appendix D of the *BASIC Language Reference*.

7

Loading and Running Programs

Since BASIC programs can be saved two ways, STORE or SAVE, two different types of files are created:

- Programs STORE'd (or RE-STORE'd to an existing file) have file type PROG, and you must `LOAD` the program into memory and then run it.
- Programs SAVE'd (or RE-SAVE'd) are stored as ASCII characters, and you must `GET` the program and then run the program.
  - Programs SAVE'd on HFS volumes are HP-UX type files.
  - Programs SAVE'd on SRM or LIF volumes are ASCII type files.



Loading and Running Files

Loading PROG Files into Memory (LOAD)

LOAD is used for programs saved with the STORE command (files of type PROG).

Using LOAD

Table 7–1 shows examples of the LOAD command.

Table 7–1. Examples of LOAD

Example	Explanation
`LOAD "Scenery"`	Clear program memory and load the PROG file `Scenery`.
`LOAD "../sample/Program"`	Clear program memory and load the file `Program` from the `sample` directory (a sub-directory of the parent directory).
`LOAD "CIRCLES",10`	Clear program memory, load the PROG file `CIRCLES`, and run the program starting at line 10.

Clearing Memory

When you LOAD a program, any program currently in memory is cleared. However, you may need to clear variables or clear the RECALL buffer (the RECALL buffer holds a stack of previously run commands). The `SCRATCH` command lets you clear various parts of memory:

`SCRATCH` clears all program lines and all variables not in COM (see *BASIC Programming Techniques* or *BASIC Language Reference* for a description of COM).

`SCRATCH A` clears the BASIC program, variables, and key definitions. The RECALL buffer remains intact.

`SCRATCH C` clears all variables in memory including those in COM .

`SCRATCH R` clears the RECALL buffer.

For more on SCRATCH, see the *BASIC Language Reference*.

Problems You Might Encounter

Here are some possible errors:

- If the file is not a PROG file, LOAD is not performed and an error 58 (improper file type) is reported.
- If the file is not found, error 56 (file not found) is reported.
- If your program was written in a version of BASIC different from the one currently installed, you receive `WARNING: PROG contains BIN with invalid revision`. You can load and run the program, but it might not run properly on your system.

For More Information

For more on LOAD, see the *BASIC Language Reference*.

Loading ASCII or HP-UX Type Files Into Memory (GET)

ASCII or HP-UX type files may contain programs that need converted to BASIC internal format. These files were saved with the SAVE command or created in the HP-UX environment (for example, with another editor—see the chapter "Running HP-UX Commands").

You must use the GET command to load these programs into memory.

Using GET

Table 7–2 shows examples of the GET command.

Table 7–2. Examples of GET

Example	Explanation
`GET "FORMULA"`	Clear program memory and bring in the contents of the ASCII or HP-UX file FORMULA.
`GET "RATES",10`	If there is no program in memory, bring in contents of ASCII or HP-UX file RATES, and run the program beginning at line 10.
`GET "RATES",250,100`	If there is a program in memory, bring in contents of ASCII or HP-UX file RATES, delete from lines 250 to the end of the current program and append RATES (renumbering it) to the end of the current program. Run the program beginning at line 100.

Problems You Might Encounter

Here are some possible errors:

- If the first line does not start with a valid line number, GET is not performed and error 68 (syntax error during GET) is reported.
- If the file is not an ASCII or HP-UX file, GET is not performed and error 58 (improper file type) is reported.
- If the file is not found on the volume, error 56 (file not found) is reported.
- If there are syntax errors in the file, error 68 is reported and the lines with syntax errors are commented.

For More Information

See the *BASIC Language Reference* for more information on GET.

Running a Program

Once you load a program into memory with LOAD or GET (note that if you load your program with GET, it could have some syntax errors since it might not have been created with EDIT—with automatic syntax checking), you can run the program.

Using RUN

Table 7–3 shows examples of running a program once in memory.

Table 7–3. Running a Program

Example	Explanation
`RUN`	Runs the program currently in memory.
f3	In the `User 1` menu there is a RUN softkey (press Shift-User).
`RUN 200`	Run the program currently in memory starting at line 200 (if there is no line 200, start at closest line greater than line 200).
`RUN Label`	Run the program currently in memory starting at `Label` (if there is no label `Label`, an error occurs).

While a program is running, the keyboard is still active under most conditions. You can:

- Stop or pause the program (see "Pausing and Stopping a Program")
- Execute commands and change variables. This feature is mainly used for troubleshooting (see "Debugging Programs" in *BASIC Programming Techniques*).

Problems You Might Encounter

If your program terminates in an error:

- If the program is a purchased application, you should reference the application's documentation or contact your sales office.
- You can edit the program and attempt to rectify the error by entering the EDIT mode (you are automatically placed at the line where an error has occurred).

For More Information

For more on RUN, see the *BASIC Language Reference.* For more on error handling, see *BASIC Programming Techniques,* "Handling Errors".

Pausing and Stopping a Program

While a program is running, you have several keys available to stop or pause the program to control program execution.

Keys to Pause and Stop a Program

Table 7–4. Pausing and Stopping a Program

Key	Explanation
Stop	Pause the execution; the program line at the bottom of the screen is the next line executed when the program is continued.
f1 (Step)	(in System menu) When program is paused, execute the next program line.
f2 (Continue)	(in System menu) Continue running the program at the next program line.
Shift-Stop	(or Ctrl-r) Stop the program (cannot be continued); maintain values of variables.
Shift-Break	(Reset) Stop the program and clear the display line; all I/O operations are aborted, any open files are closed, all interface cards are reset.
Break	(or Ctrl-c) Abort any I/O statement and pause the program. This can be used when the computer tries to output to a device that is powered down.

Listing a Program

Besides running and editing a program, you can list the program lines.

Using LIST

Use LOAD or GET to place a program in memory.

Table 7-5. Listing a Program

Example	Explanation
`LIST`	List the program in memory
`LIST 100,200`	List the program in memory from line 100 through 200.
`LIST 1850`	List the program in memory from line 1850 to the end.
`LIST Rocket`	List the program in memory from the label `Rocket` to the end.

Stopping Long Listings

Break halts long lists and returns your command line (no keyboard command pauses the list). Chapter 3 shows how to pause window output.

Sending the LIST Output to the Printer

To send a program list to the default printer (printer designated by your system administrator), run the command:

```
PRINTER IS "| lp"
```

When you `LIST` or `CAT` after this command, the output is sent to the default printer, but it is not printed until you "turn off" this option, with:

```
PRINTER IS CRT
```

For More Information

See the *BASIC Language Reference* for more on the LIST and PRINTER IS commands.

Preventing Programs From Being Listed

If you have a program you don't want others to see, you can prevent the entire program or certain lines from being listed or edited.

Using SECURE

Table 7–6 shows you how keep other users from viewing program lines. SECUREd program lines are replaced with a "*" when you try to LIST or EDIT them. Once you execute a SECURE command on a file, it *cannot be un-secured.* You may wish to keep an un-secured back-up copy of all programs.

Table 7–6. Protecting Program Listings

Example	Explanation
`SECURE 30,60`	Protects lines 30 through 60 in the current program from being listed
`SECURE`	Protects entire program from being listed

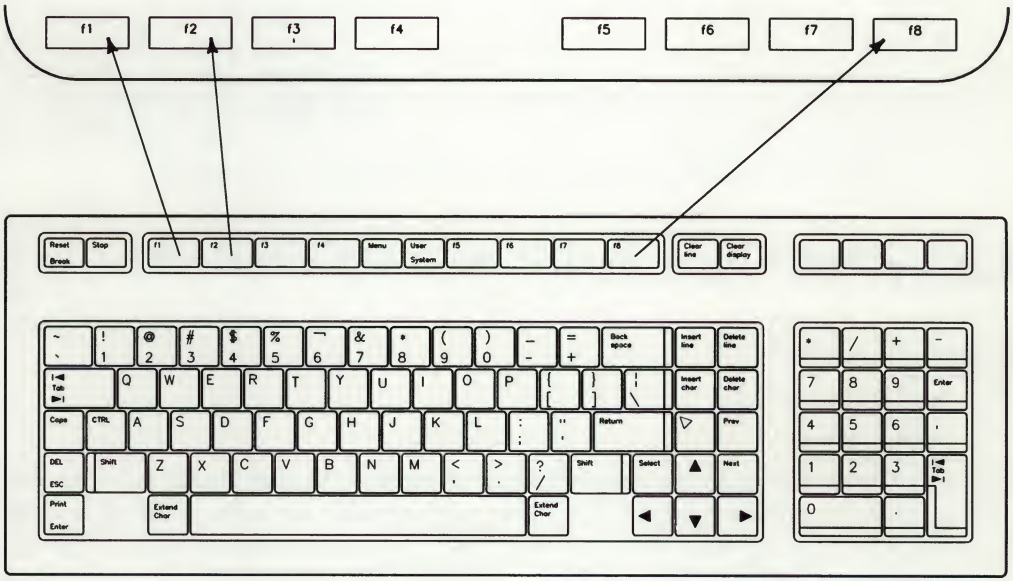
For More Information

See the *BASIC Language Reference* for more details on SECURE.

8

Using Softkeys

This chapter shows you how to use your system's softkey menus and how to create and use your own softkeys.



Mapping Function Keys to Softkey Labels

Accessing Softkeys

Your keyboard has a row of keys called **function keys.** On the ITF keyboard, these keys are along the top of the keyboard labeled [f1] through [f8]. Each key has a function defined by a **softkey definition** so that when you press the function key, it performs some computing action.

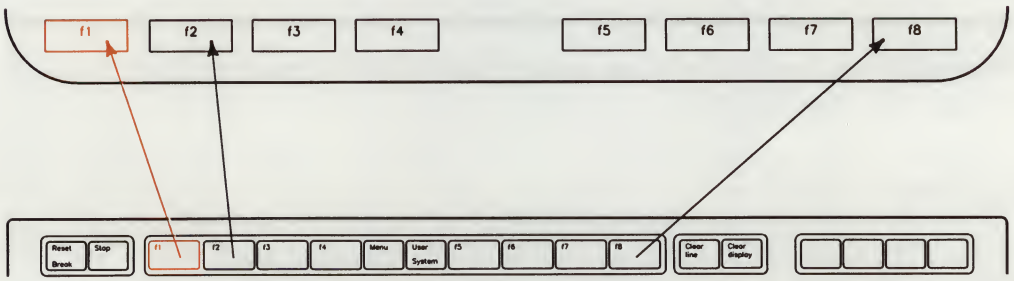


Figure 8–1. How Softkeys and Function Keys are Related

Turning On Your Softkey Labels

If you do not see softkey labels on the bottom of your screen (or window), press [Menu] or type:

```
KEY LABELS ON [Return]
```

A series of highlighted boxes appear on your screen. To use a softkey, scan the softkey labels on the screen for a command or function you wish to use and press the corresponding function key. For example, the first softkey label on the left corresponds with [f1], the second [f2], and so on.

Problems You May Encounter

If you don't see the softkey labels after you execute the KEY LABELS ON command, press the [Menu] key (this toggles the softkey labels on and off).

Accessing Softkey Menus

The first softkey menu (the row of softkey labels) you see after the KEY LABELS ON command is not the only softkey menu available. There are four menus with many different functions available.

Cycling Through Softkey Menus

Table 8–1 shows you how to access the softkey menus.

Table 8–1. Accessing Softkey Menus

Task	How to Perform Task
Turn on screen labels	Press Menu or type: `KEY LABELS ON` Return
See the SYSTEM menu	Press System or type `SYSTEM KEYS`
See the USER 1 menu	Press User (Shift-System) or type `USER 1 KEYS`
See the USER 2 menu	Press Shift-Menu or type `USER 2 KEYS`
See the USER 3 menu	Press Shift-Menu from the USER 2 menu or type `USER 3 Keys`
Cycle through the USER menus	Press Shift-Menu

Using the System Softkeys

The next four pages show diagrams of each menu (SYSTEM, USER 1, USER 2, USER 3) and how to use each softkey.

Using the SYSTEM Menu

This menu appears first after you boot up or run the `KEY LABELS ON` command.

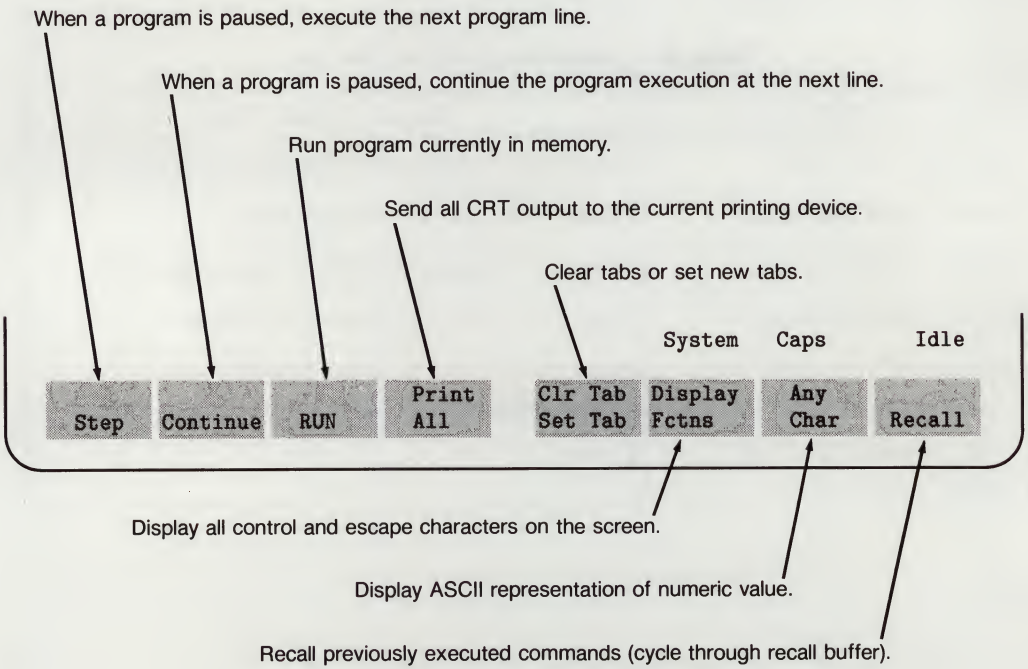
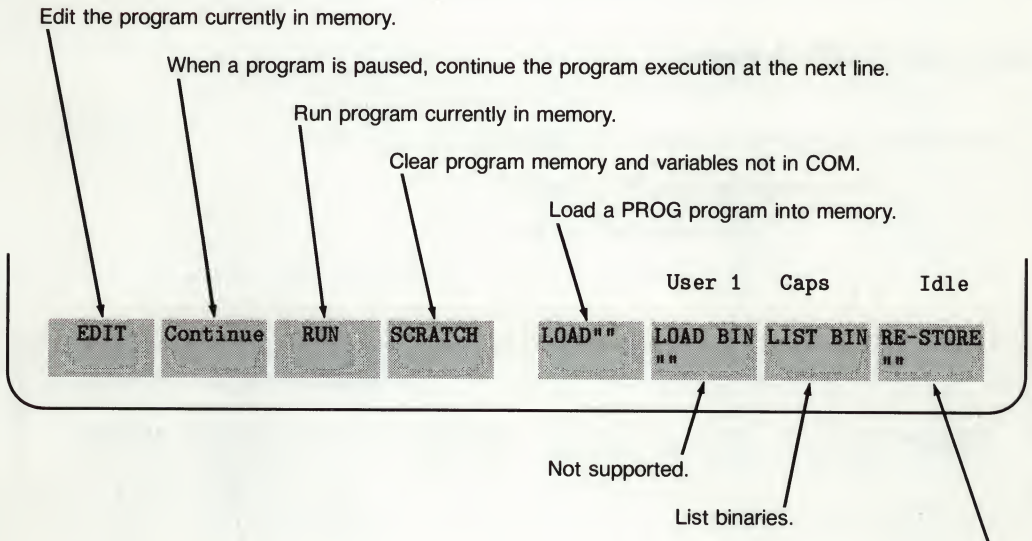


Figure 8–2. SYSTEM Softkey Menu

Using the USER 1 Menu

[User] ([Shift]-[System]) brings up the USER 1 menu.



Create a file and store program or softkey definition. If file exists on system, the old file is removed and the new file stored. If file does not exist, it is the same as a STORE command.

Figure 8-3. USER 1 Softkey Menu

Using the System Softkeys (Continued)

By using the key [Shift]-[Menu] you cycle through the USER menus.

Using the USER 2 Menu

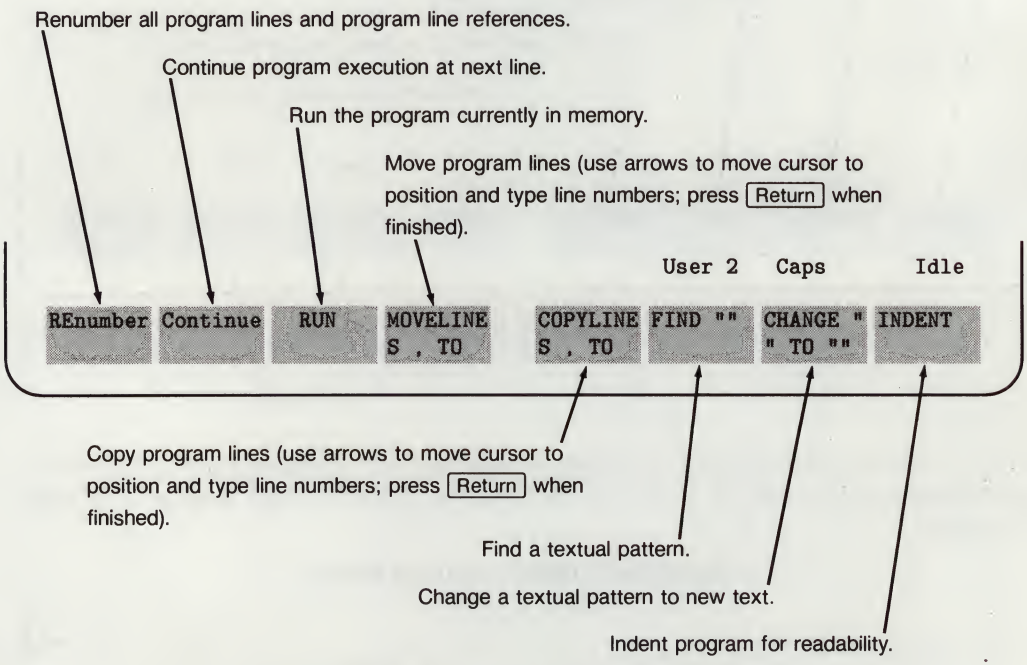


Figure 8-4. USER 2 Softkey Menu

Using the USER 3 Menu

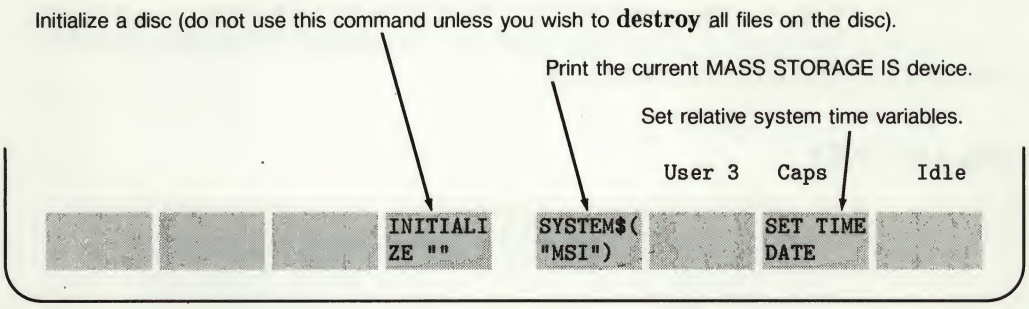


Figure 8-5. USER 3 Softkey Menu

Listing Softkey Definitions

You have the ability to redefine the softkey definitions for your own use (described in "Redefining Softkeys"). You may wish to see a list of the current softkey definitions to help you define your own.

Using LIST KEY

To list the current softkey definitions (for the menu currently in use), type:

```
LIST KEY Return
```

This shows you what the softkey definitions look like. The list is long, so you might send the list to the printer with:

```
LIST KEY #PRT Return
```

Where PRT is the default printer at address 701 (to send the list to a file if you wish to use HP-UX commands to view the file, type:

```
PRINTER IS "| cat >> tempfile"
```

and when you type `LIST KEY`, the output goes to the file `tempfile`. See PRINTER IS in the *Language Reference*.

The printed list shows the characters `System key:` in place of control characters that most printers cannot print. For example, you would see a list like:

```
Key 1:
System key: #
EDIT

Key 2:
System key: C

Key 3:
System key: R
 ...
 ...
```

For More Information

See the *BASIC Language Reference* for more on LIST KEY (LIST reference page).

Redefining Softkeys

Redefining softkeys with your own softkey definitions is done with the EDIT KEY command. Softkeys are numbered 1 through 24 (1-8 are the USER 1 keys, 9-16 are USER 2 keys, and 17-24 are USER 3 keys). The examples that follow show several ways to edit softkey definitions.

Memory Available for Softkey Definitions

The system uses about 1024 bytes of memory to store typing-aid softkey definitions. Each definition can have:

- up to 256 characters on systems with high resolution displays
- up to 160 characters on systems with medium resolution displays.

Exceeding these limits results in lost characters.

Restoring the Softkey Definition to Power-Up Defaults

To restore the power-up default definitions, execute the following statement without specifying a file:

```
LOAD KEY [Return]
```

Example: Re-define Softkey [f1]

This example shows you how to redefine softkey [f1] to print the characters: `My own softkey`.

1. Enter the edit-softkey mode for [f1] by typing: `EDIT KEY 1` [Return]

   The system displays the key's current definition (on the keyboard input line), followed by a message to indicate you can now modify the definition.
2. Clear the key's current definition by pressing: [Shift]-[Clear line]
3. Enter the new definition; type: `My own softkey`
4. Exit the softkey edit mode and update the softkey; press: [Return]

   (If you don't want to update this softkey's definition, press [Stop] to abort.)
5. Press [f1] to see if your new definition works. (If it doesn't, try the process again.)
6. Press [Shift]-[Clear line] to clear the line for the next example.

Example: Re-define Softkey [f2]

This example defines a softkey that clears the line you are on, types a command, and then executes that command (this is how the RUN and CONTINUE softkeys are defined by the system).

1. Enter the edit-softkey mode (here's another way):
    a. Get the USER 1 menu (Press [Shift]-[System])
    b. Press [f1] (EDIT softkey)
    c. Press [f2] (watch how the system prints `EDIT KEY 2` for you)
    d. Press [Return]
2. To clear the current softkey definition, press: [Shift]-[Clear line]
3. Press: [CTRL]-[Shift]-[Clear line]

    This prints a character that represents the "CLEAR LINE" character: #. When the system executes this softkey, it first clear the keyboard input line.
4. Type: `LIST` [CTRL]-[Return]

    The [CTRL] key lets you place characters such as [Return] in the softkey definition. Notice the character `E` on the screen.
5. To save the softkey definition, press: [Return]
6. To see if your softkey works, press: [f2]

This softkey executes the `LIST` command. If you have a program currently in memory, it is listed. Also, notice that your softkey label for [f2] contains an extra character in the label. The next example shows you how to clean up the softkey labels.

Redefining Softkeys (Continued)

Example: Cleaning up a Softkey Label

This example shows you how to clean up the [f2] label for your LIST command used in the previous example.

1. Type: EDIT KEY 2 [Return]
2. To clear the current softkey definition, press: [Shift]-[Clear line]
3. Type the following line (there are 12 spaces after LIST):

   LIST [CTRL]-[Shift]-[Clear line] LIST [CTRL]-[Return]

4. Press [Return] to save.
5. Press [f2] to see how the new definition works. Notice that LIST is momentarily displayed, then cleared and displayed again. The first LIST is not executed.

For More Information

See EDIT KEY in *BASIC Language Reference*.

Defining Softkeys that Require Inserting Text

Some softkeys not only type a command but place the cursor so you can insert within the command.

Looking at the MOVELINES Softkey

Let's take a look at the MOVELINES softkey on the USER 2 menu.

1. Get to the USER 2 menu (Shift-System or USER 2 KEYS)
2. Type: EDIT KEY 12 Return

You should see a line like:

```
#MOVELINES , TO    < % < < < < < +
```

- # represents CTRL-Shift-Clear line
- % represents CTRL-Clear line
- < represents the left arrow (CTRL-◄)
- + represents the insert character (CTRL-Insert char).

When you press this key, the cursor is placed before the comma and puts you in the insert mode. This way you can type the line numbers. To move the cursor after the comma or to the end of the line (after TO), use the right arrow key (►).

Creating a Softkey Definition File

If you have modified any of the softkeys, you can store the definitions in a file and thus use them in subsequent sessions.

Storing Softkey Definition Files

To create a file to store the current softkey definitions, type this command:

```
STORE KEY "MyKeys" Return
```

If the file already exists, you must use this command:

```
RE-STORE KEY "MyKeys" Return
```

Loading Softkey Definitions into Memory

To load a softkey definition stored in a file, type:

```
LOAD KEY "MyKeys" Return
```

Where `MyKeys` is a softkey definition file you previously created.

For More Information

You can also write a program that defines the softkeys using the SET KEY statement in a BASIC program. See the "Communicating with the Operator" chapter of *BASIC Programming Techniques* for details.

9

Using HP-UX Commands in BASIC/UX

The EXECUTE command is used to run HP-UX commands while in BASIC/UX. This chapter also discusses how to localize BASIC/UX error messages in other languages.

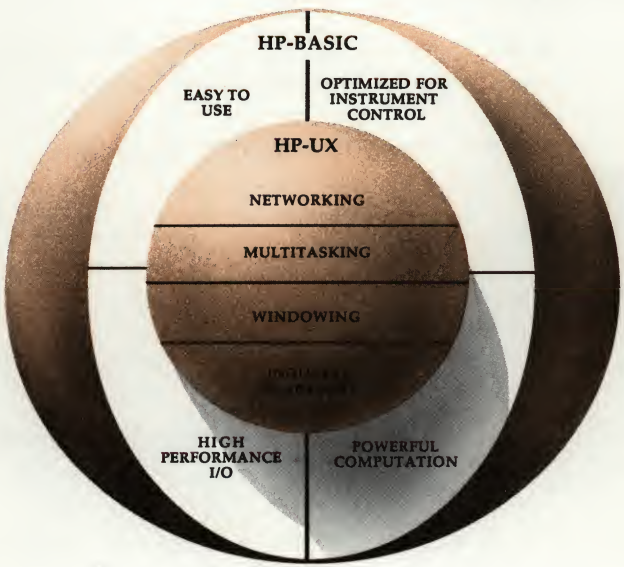


Figure 9-1. HP BASIC/UX in Relation to HP-UX

Using the EXECUTE Command

EXECUTE runs an HP-UX command in a "sub-shell". This means any HP-UX variables created while the EXECUTE command is running are erased when control is returned to BASIC/UX.

Prerequisites

In X Windows, the output from EXECUTE is displayed in the HP-UX (root) window that initiated BASIC/UX (if you boot directly into BASIC/UX, there is a window "underneath" your BASIC/UX window; to see the window, move the pointer to the root window, press [Extend char] and hold it down, and press the *right* button).

Examples of the EXECUTE Command

The following table shows some examples of the EXECUTE command. For a detailed look at EXECUTE and its options, see the *Language Reference*.

Table 9-1. Example EXECUTE Statements

Example	Description	Example Results
`EXECUTE`	Run HP-UX in the window from which BASIC/UX started. Type `exit` [Return] to return to BASIC/UX.	`$`
`EXECUTE "date"`	Runs the HP-UX command `date`. Type `date` in lower-case.	`Tue Apr 5 09:55:53 MDT 1988`
`EXECUTE "more myfile"`	Run the HP-UX command `more` which shows the contents of `myfile`.	`This is a file which is not a PROG file. It could have been created in HP-UX.`

How to Run HP-UX Commands in the Background

If you have an HP-UX command that:

- takes a long time to execute
- is a concurrent process (runs all the time),

you can place it in the "background" so it runs at the same time you use BASIC/UX.

To run multiple HP-UX processes, add the ampersand (&) character after the HP-UX command in the BASIC/UX statement. For example,

```
EXECUTE "ps &"
```

executes the HP-UX command `ps` and lets you continue executing BASIC/UX commands. If you omit the &, you cannot execute any more commands until the process is completed. Without the &, you can halt the process (have the pointer in the root [HP-UX] window if in X Windows) with:

- Break
- CTRL-c
- `exit` Return

If in X Windows and the process continues, move the pointer to the BASIC/UX window and press Reset (Shift-Break). Use this as a last option as it may clutter the system with parts of the killed process.

You cannot run `rmb` in the background (it serves no purpose, and uses system resources; therefore, an error message is generated).

Using the EXECUTE Command (Continued)

How EXECUTE Displays

When you are in X Windows, the EXECUTE command runs in the window that started BASIC/UX (root window).

If you are not in X Windows, the screen is cleared and the EXECUTE command output shown on a new screen.

Table 9-2. EXECUTE Not in X Windows

If you want to ...	Do this ...
Return to BASIC/UX	When the process completes, you are prompted to PRESS ANY KEY TO CONTINUE:. When you press a key, the screen is cleared, and you are returned to BASIC/UX.
Stop a process while it is running	Press [Break].
Leave an HP-UX shell	Type `exit` [Return].

If in X Windows and your process does not stop after using the above methods, move the pointer to the BASIC/UX window and press [Reset] ([Shift]-[Break]).

EXECUTE Runs as a Child Process

When you run an EXECUTE command, it is processed as a **child** process. In other words, the command defines its variables only for the time the command is being processed. When you return to BASIC/UX, those variables are lost.

You cannot run a networking set-up command with EXECUTE because the connection is lost once you return to BASIC/UX. See "If You Have Networking Options" for details.

For More Information

See *BASIC Language Reference* for details and other options of `EXECUTE`, for the `SAVE ALPHA OFF` option which saves graphic output, and `RETURN` variable which allows you to return to BASIC without pausing.

See *A Beginner's Guide to HP-UX* for general information on operating in HP-UX. For details on HP-UX commands, see the *HP-UX Reference*.

Using Some HP-UX Commands and Utilities

The commands in Table 9–3 can be used in the EXECUTE statement. If you wish to try these commands, you should reference *A Beginner's Guide to HP-UX* for any detailed information or a complete list of beginning HP-UX commands.

Table 9–3. Some Useful HP-UX Commands

Command	Description
`cal`	Display a calendar for the current month.
`date`	Display the current date and time.
`hostname`	Display your system's "node" name.
`mailx` *user*	Send an electronic mail message to *user*.
`man` *command*	Display on-line information on *command*.
`whoami`	Display your user name.
`cp` *file newfile*	Copy *file* to *newfile*.
`ls`	List (catalog) files in current directory.
`mkdir` *dir*	Create a directory named *dir*.
`more` *file*	See the contents of *file*. Press the spacebar to continue scrolling when content list pauses.
`mv` *file newfile*	Change the name of *file* to *newfile* (`mv` can also be used to move a file to a specified directory).
`pwd`	Display the path name of the current directory.
`rm` *file*	Destroy *file* (file is irretrievable).
`rmdir` *dir*	Destroy the directory *dir* (cannot contain files).
`ps -ef`	Display all current processes running on the system.

For example, `EXECUTE "date"`

If You Have Networking Options

If your system is set up to network to other systems (see your system administrator), you must run the networking commands in HP-UX *before* you enter BASIC/UX.

For example, if your system is set up to run the `netunam` networking command:

1. Make sure your system is in HP-UX (`QUIT` from BASIC/UX; do not use the `EXECUTE` command).
2. Run the command, for example:

   `netunam /net/`*system_name user_name*`:`

   When it prompts you for the password, type the other system's password and Return.
3. Enter BASIC/UX (`rmb`).

As mentioned above, `EXECUTE` runs the HP-UX command as a child process. So, if you run a networking command with `EXECUTE`, the networking connection is lost once you return to BASIC/UX.

Constraints

BASIC/UX expects the network special file to be located in the `/net` directory. If you wish to use a network special file in another directory, you must link a file in `/net` to the desired location (see "Linking Files" in Chapter 5). If this is not done, unexpected results can occur.

For More Information

See the *Network Services/9000 LAN Node Manager's Guide* for more on `netunam`.

Using BASIC/UX Program Files with HP-UX Commands (and Visa Versa)

Only HP-UX type files can be used with HP-UX commands and utilities. This section shows how to convert programs stored in ASCII or PROG files to HP-UX file types.

BASIC/UX creates HP-UX file types when a SAVE or RE-SAVE is done to an HFS volume. SAVE and RE-SAVE create ASCII file types when done to a LIF or SRM volume.

Making ASCII Type Files HP-UX Type Files

While you are in BASIC/UX,

1. Load the program (file) into memory (if it isn't already):

   LOAD "*file*" Return

   where *file* is the name of the file.
2. Type:

   SAVE "*newfilename*" Return

   to properly save *newfilename*.
3. Type:

   CAT Return

   to see that *newfilename* has file type HP-UX.

If you wish to use the file with an HP-UX command (for example, the `vi` editor), you'll need to exit BASIC/UX (or create a sub-shell with `EXECUTE`) first.

Using HP-UX Files in BASIC/UX

Files of file type HP-UX (created by HP-UX; in an HP-UX editor, for example) can be use by BASIC/UX. You must use `GET` to bring the file into memory. Make sure the file contains characters for a BASIC program or you have trouble EDITing it.

Problems You Might Encounter

If you are attempting to `GET` an HP-UX file that does not contain a BASIC program, you receive an error message like, `ERROR 68 Syntax error occurred during GET`. You cannot edit the file in BASIC.

Converting Error Messages to Another Language

HP-UX Native Language Support (NLS) tools let you localize BASIC/UX error messages to languages other than English.

Prerequisites

- You must be logged in as `root` or be super-user (`su`) to modify the message file. See your system administrator to do this task.
- The fileset `NCORE` from partition `NLS` must be installed on your system. To check,

  ```
  lsf /etc/filesets | grep NCORE Return
  ```

  If you see `NCORE` returned, it means you have the fileset loaded. If you see nothing but a new prompt, you must load the fileset. See the *System Administrator Manual,* "Customizing the HP-UX System" chapter, "Installing Optional Software and Updating Your HP-UX Kernel". You must be system administrator to do this task.
- You should know the language to which you are converting the messages.
- You should be familiar with C language `printf` statements.
- You must be familiar with an HP-UX editor (such as `vi`).

Converting BASIC/UX Error Messages

1. Login as `root` or become super-user (`su`).
2. Copy a default file to `/tmp` as a working file:

   ```
   cp  /usr/lib/rmb/newconfig/rmb.msgs  /tmp/rmb.msgs
   ```

3. Change directories: `cd /tmp`
4. Edit `rmb.msgs` (for example, with `vi`) and change the English strings to the appropriate language equivalents.
   - Do not change lines beginning with `$`
   - Do not change or remove any numbers.
   - Do not remove any of the format information (for example, quoted strings are C programming language print control statements).

   You may wish to print out the file first for viewing as it is a large file.
5. Execute the HP-UX `gencat` utility:

   ```
   gencat rmb.cat rmb.msgs
   ```

6. List the `/usr/lib/nls` directory to choose the language catalog where you store the new message file:

   ```
   ls /usr/lib/nls
   ```

7. Move the new file to the appropriate directory in `/usr/lib/nls`:

   ```
   mv rmb.cat /usr/lib/nls/language
   ```

   For example, if you are converting to Spanish,

   ```
   mv rmb.cat /usr/lib/nls/spanish
   ```

Converting Error Messages to Another Language (Cont.)

Problems You Might Encounter

Be sure to include spaces between the file names in the above commands. If you receive a message like

```
mv [-f] f1 f2
mv [-f] f1 ... fn d1
mv [-f] d1 d2
```

while running the `mv` command, it means you forgot the spaces.

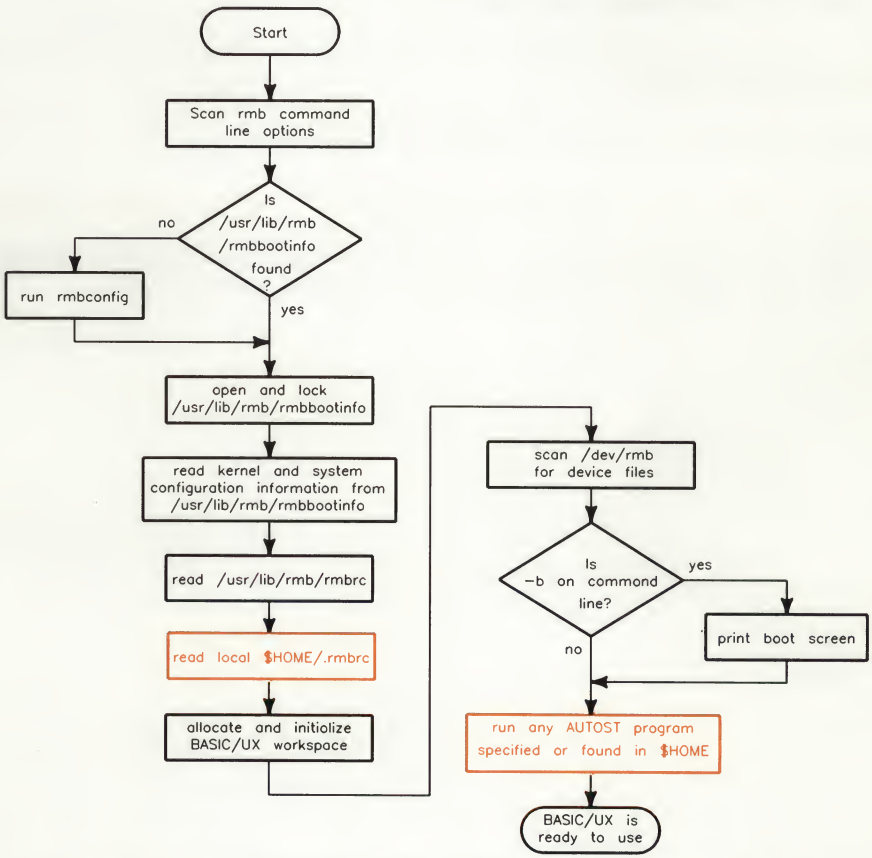
For More Information

- *HP-UX Reference* for `gencat(1)`.
- *HP-UX Concepts and Tutorials: Device I/O and User Interfacing,* "Native Language Support" section.

10

Creating Environment and Autostart Files

An environment file sets system variables when you start BASIC/UX. An autostart file then is automatically run with commands you specify.



When Autostart Files are Read During HP BASIC/UX Boot Time

Customizing BASIC/UX Sessions

When you start a BASIX/UX session, the system looks for a file:

```
/usr/lib/rmb/rmbrc
```

to set the default environment (the environment consists of system variables that affect how the system performs some tasks).

A template environment file:

```
/usr/lib/rmb/newconfig/rmbrc
```

is available for you to customize and move to your home directory and call `.rmbrc` (lower-case, preceed with .).

For example, copy the default file to `/users/leslie`:

```
COPY "/usr/lib/rmb/newconfig/rmbrc" TO "/users/leslie/.rmbrc"
```

Then EDIT the file and make any necessary changes based on the descriptions that follow. See "How to Create Your Environment File" for details.

What Variables Can Be In The Environment File?

More details about these variables follow the table.

Table 10–1. Global Environment Variables

Name of Variable	Range of Values	Default Values	Description
`autostart`	*pathname*	n/a	Pathname of an autostart file.
`errormode`	`on`, `off`	`on`	Generate error messages for BASIC Workstation, BASIC/UX compatibility.
`graphics_buffer`	`on`, `off`	`on`	Turn on graphics buffering to speed up graphics processing time.
`hfs_buffer`	`on`, `off`	`on`	Turn on HFS file system buffering.
`plock`	`all`, `t`, `d`, `w`	n/a	Lock text area, data area, workspace or all of BASIC/UX into memory (any combination of `t`, `d`, and `w` is valid).
`rmb_shmem_addr`	2m to 16m	`shmmaxaddr`	Upper address for BASIC/UX shared memory.
`rtprio`	0 to 127	n/a	Set real-time priority level (used only by root or privgroup members).
`workspace`	64k to 6m	1m	Size of BASIC/UX workspace (integer values only).

Customizing Your BASIC/UX Session (Continued)

Here is more detail on the environment variables for BASIC/UX, as well as some additional statements you can add to `rmbrc`.

Running an Autostart Program (autostart)

If you wish an autostart program, you can specify the file with `autostart`. For example, using the program, `/users/leslie/AUTOST`, you would enter:

```
50 !autostart=/users/leslie/AUTOST
```

into the `.rmbrc` file. The line number is arbitrary, and the example shows a file created using EDIT mode.

Generate Compatibility Error Messages (errormode)

If you port programs created on BASIC Workstation systems, you may have some errors with, for example, commands not supported on BASIC/UX (such as `LOAD BIN`).

- `60 !errormode=on` has error messages generated.
- `60 !errormode=off` does not print error messages.

The above examples show an arbitrary line number created using EDIT mode.

Graphics Buffering (graphics_buffer)

When using graphics, you can choose to have graphics buffering:

- `70 !graphics_buffer=on` turns on graphics buffering. The image is faster than when off, but it could be choppy when an image moves on the screen.
- `70 !graphics_buffer=off` is slower than when on, but the image is smoother when moving on the screen.

The above examples show an arbitrary line number created using EDIT mode.

HFS File System Buffering (hfs_buffer)

This variable determines how data is written to a disk:

- `80 !hfs_buffer=on` saves data in a buffer and writes to a disk periodically. This makes system operations faster, but could cause a greater amount of data to be lost if a power failure occurs, or if the system is improperly shut down.
- `80 !hfs_buffer=off` writes the data to the buffer, then immediately to the disk. This causes the performance of OUTPUT to be much slower.

The above examples show an arbitrary line number created using EDIT mode.

Locking BASIC/UX in Memory (plock)

To lock the text area, data area, workspace, or all of BASIC/UX into memory (disables swapping),

- `90 !plock=all` locks BASIC/UX into memory.
- `90 !plock=t` locks text area into memory.
- `90 !plock=d` locks data area into memory.
- `90 !plock=w` locks workspace into memory.

Any combination of `t`, `d`, or `w` is valid (for example, `plock=td`). The above examples show an arbitrary line number created using EDIT mode.

Shared Memory Address for BASIC/UX (rmb_shmem_addr)

Set the upper address for BASIC/UX shared memory with `rmb_shem_addr`. For example,

```
100 !rmb_shmem_addr=16m
```

sets the upper limit at 16M. The default is the HP-UX variable `shmmaxaddr` (set at 16M). The above example show an arbitrary line number created using EDIT mode.

Setting Real-Time Priority (rtprio)

Set real-time priority (values 0 to 127, where 0 is highest priority):

```
110 !rtprio=50
```

The above examples show an arbitrary line number created using EDIT mode. These values can only be used by root or privgroup members. To let others use real-time priority, see the `setprivgrp` command in the *HP-UX Reference*.

Setting Size of BASIC/UX Workspace (workspace)

Set the size of BASIC/UX workspace, for example:

```
120 !workspace=2m
```

The above example show an arbitrary line number created using EDIT mode. Your value should be dependent on the size of programs to be run. For example, a program with lots of subprograms, CSUBs, etc., needs more workspace than a small program.

Setting Up Automatic Device File Locking and Mapping

Set up automatic locking, memory mapping, or set `io_burst` on an I/O interface. Note that `autolock` + `automap` is equal to `autoburst`.

Determine the select code of the interface, and include this in the `.rmbrc` file:

`interface` *select_code*; *option*

where *option* is one of the following:

- `autolock`
- `automap`
- `autoburst`
- `normal`.

Mapping BASIC Mass Storage Volume Specifiers to HFS Directories

If you have programs that access mass storage devices with the volume specifiers, you can map a volume specifier to an HFS directory with this entry:

```
disk scba,volume,unit = directory name
```

where

- *scba* is: select code * 100 + bus address
- *,volume* is the volume number (optional; use the comma if you include this)
- *,unit* is the unit number (optional; use the comma if you include this)
- *directory name* is an HFS directory. This could be the directory under which the disk is mounted.

For example, map a device on select code 7, bus address 2, volume 1 mounted under `/disk1`.

```
disk 702,1 = /disk1
```

How to Create Your Environment File

To make the `.rmbrc` file compatible with HP-UX, you must either create a new `.rmbrc` file and place it in your HOME directory, or copy the system default and modify it. There are two ways to format the file:

- Using an HP-UX editor (`vi`, for example).
- Using the BASIC/UX editor.

Using HP-UX Editor

1. Be in HP-UX (`QUIT` from BASIC/UX).
2. Copy the default `rmbrc` file:

```
cp /usr/lib/rmb/newconfig/rmbrc $HOME/.rmbrc
```

3. Modify the `.rmbrc` file, for example: `vi .rmbrc`

Preceed comments with the `#` character; *include statements without spaces between variables and values.* For example:

```
# This is a sample rmbrc file edited in HP-UX
interface 7;autolock
errormode=off
workspace=1m
# End of sample rmbrc file
```

Files created with an HP-UX editor cannot be modified using the BASIC Editor.

Using the BASIC Editor

1. Enter BASIC/UX: `rmb`
2. Make sure you are in your HOME directory (`EXECUTE "echo $HOME"` tells you your home directory).
3. Copy the default file:

   ```
   COPY "/usr/lib/rmb/newconfig/rmbrc" TO ".rmbrc"
   ```

4. Edit the new file: `EDIT` and make modifications (see below for an example file).
5. Same the file: Shift-Clear line `RE-SAVE ".rmbrc"`

All lines must be preceded with line numbers and an exclamation mark. Precede comments with `!#` or `REM`. For example:

```
10 REM This is a sample rmbrc file edited in BASIC
20 !interface 7;autolock
30 !errormode=off
40 !workspace=1m
50 !# End of sample rmbrc file
```

Creating an AUTOST File

Autostart files in BASIC/UX are similar to those in Workstation BASIC 5.1. See the figure on the first page of this chapter for how the system searches for an autostart file.

You can also indicate the autostart file in the `rmb` command (see the *HP-UX Reference*).

Why AUTOST Files Are Used

If you must perform the same task(s) each time you enter BASIC/UX, you can do the task(s) automatically with an autostart file. An autostart file is really a program that is run at boot time. You can LOAD files, PRINT statements, RUN programs.

An Example AUTOST File

The following is a sample AUTOST file. It greets the user and then loads and runs a program.

```
10   PRINT "Welcome to the System."
20   PRINT
30   INPUT "Press RETURN to start the MONITOR program",C$
40   LOAD "MONITOR",1        !load and automatically run MONITOR
50   END
```

When you press Return as prompted, the file `MONITOR` is loaded and run.

To edit you own AUTOST program, enter the EDIT mode and create a program that fits your needs. Then SAVE the file in your home directory, named `AUTOST`.

Problems You Might Encounter

Be sure to debug your AUTOST program before you re-enter BASIC/UX. This avoids unnecessary errors.

For More Information

See the *HP-UX Reference* for information on the `rmb` command options (shows how to specify your own AUTOST file).

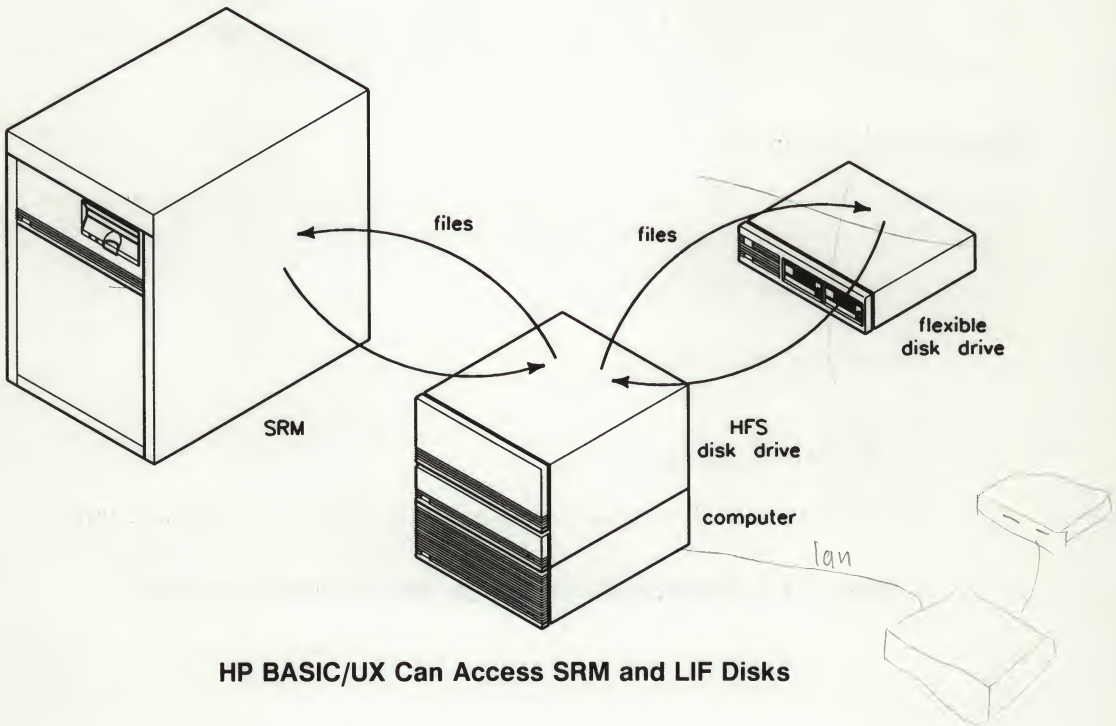
Notes

11

Working with LIF Disks and the SRM

Most of your file operations can be performed using the system's hard disk. However, you may wish to use HP BASIC programs that reside on flexible disks or an SRM (Shared Resource Manager), or use flexible disks for storing files.

Note: Your system administrator must set up the disk device or SRM on your system before you can access it.



HP BASIC/UX Can Access SRM and LIF Disks

Copying Files From Other Disks

If you need to copy files from a flexible disk or a tape, you must determine its **msvs** ("Mass Storage Volume Specifier").

Prerequisites

See your system administrator for a device's mass storage volume specifier.

What Volumes Are

A volume is usually a disk that is formatted either HFS (Hierarchical File System; your BASIC/UX hard disk is formatted HFS) or LIF (Logical Interchange Format; most flexible disks are formatted LIF).

LIF is similar to HFS, but LIF volumes contain only one directory. When you connect a device to your system, you need a way to access it.

Specifying an MSVS

When your peripheral devices are added to your system, your system administrator should place a label with the msvs on each device.

The msvs takes the form, for example:

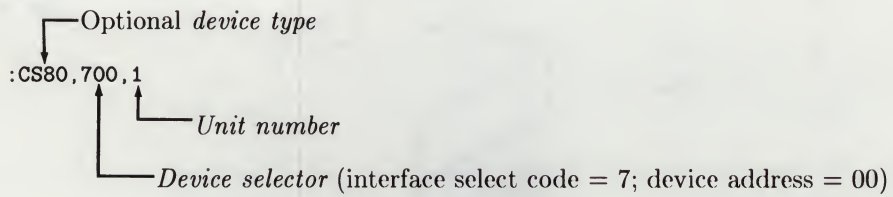


Figure 11-1. Example Mass Storage Volume Specifier (msvs)

- **Device-type** is a series of characters that identify the type of device (like a flexible disk).
- **Device-selector** is a number associated with a particular disk or tape drive,
- **unit number** is the number of the unit on the device.

If you need more information or need to determine the msvs manually, see the *Peripheral Installation Guide,* "Msvs/Unit Numbers/Minor Numbers" chapter.

Table 11–1. Example Statements with msvs

Example	msvs	Explanation
`LOAD "AFILE:CS80,700"`	`:CS80,700`	Load the file `AFILE` from the volume ":`CS80,700`"
`CAT ":,700"`	`:,700`	Catalog the volume ":`CS80,700`" (the `CS80` can be omitted; it is not required).
`COPY "AFILE:,700" TO "AFILE"`	`:,700`	Copy the file `AFILE` from the volume `:CS80,700` to the current directory (and volume).
`SYSTEM$("MSI")`	n/a	Tells you what your current mass storage device is.

Some Problems You Might Encounter

`COPY` can copy entire volumes from one LIF device to another LIF device. Be aware that it destroys the current contents of the volume to which you are copying.

For More Information

See the *Peripheral Installation Guide,* "Msvs/Unit Numbers/Minor Numbers" chapter, "Determining Your BASIC Msvs" section for more information on Mass Storage Volume Specifier.

Initializing a LIF Disk

If your system administrator has set up a LIF disk to your system, you can clear a disk or format it with the INITIALIZE command. Formatting a disk:

- prepares the disk for use by BASIC/UX
- places a LIF (Logical Interchange Format) directory on the disk
- destroys anything that was on the disk.

Prerequisites

See your system administrator for the mass storage volume specifier.

Formatting with INITIALIZE

WARNING: this command will destroy the contents of your LIF disk.

Write-Enable the Disk

For a 3.5 inch disk, move the tab to the indicated position:

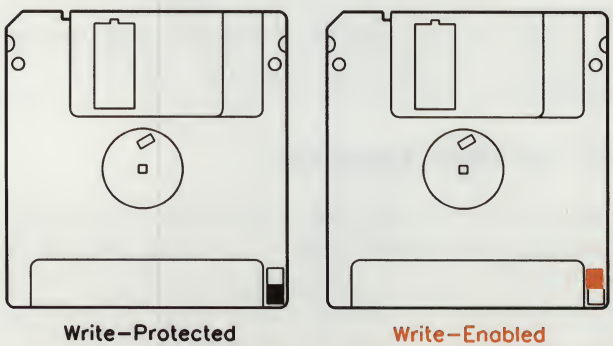


Figure 11–2. Write Enable a 3.5 inch, Double-Sided Disk

For a 5.25 inch disk, remove any cover over the notch.

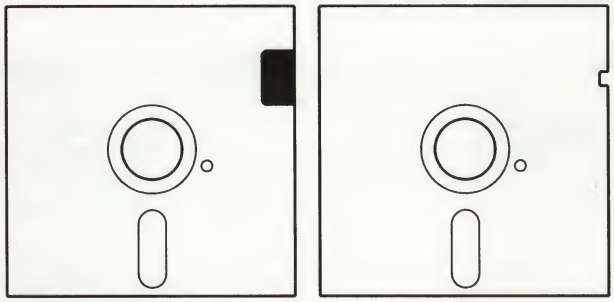


Figure 11–3. Write Enable a 5.25 inch Disk

Does the Disk Need Formatting?

1. Insert the disk into the LIF drive.
2. Determine if the flexible disk needs formatting. For example, `CAT ":,700"`.

Table 11–2 shows how to determine if the disk needs formatting using this example.

Initializing a LIF Disk (Continued)

Table 11–2. Determining If the Disk Needs Formatting

What you might see ...	What it means ...
`:CS80, 700` `VOLUME LABEL: B9836`	Your disk is already formatted LIF. You don't need to format the disk.

`:CS80, 700`
`LABEL: MyVol`
`FORMAT: HFS`
`AVAILABLE SPACE: 60168`
	Your disk is HFS format. Only format if you want to change the format to LIF.

`Error 185 HFS volumes must be mounted`	The drive is an HFS volume. You must mount the disk first (see *Installing and Maintaining the BASIC/UX System*). Only format if you want to change the format to LIF.
`Error 85 Media uninitialized`	The disk is not formatted. Continue with the steps below.
`Error 84 Record not found`	The disk is not formatted. Continue with the steps below.

Formatting the LIF Disk

Here's an example statement. In general, both the format option and interleave values can be omitted, and appropriate default values are used. See the INITIALIZE statement in the *BASIC Language Reference* for details.

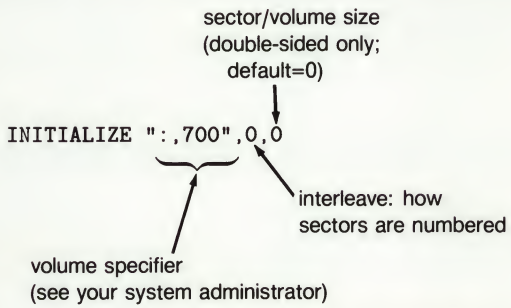


Figure 11-4. The INITIALIZE Statement

Verifying the Format

Catalog the newly formatted disk, for example:

```
CAT ":,700"
```

Here are the typical headings for a CAT on a LIF disk:

```
:,700,1
VOLUME LABEL: B9826
FILE NAME PRO  TYPE  REC/FILE  ADDRESS  TIME  DATE
```

Copying From One Flexible Disk to Another with Two Flexible Disk Drives

If you have two flexible disk drives, you can copy individual files or entire LIF volumes from one disk to another. Simply verify that the disks are both formatted and use the COPY command.

Prerequisites

Write-protect your source disk (the first disk containing the file or files to be copied).

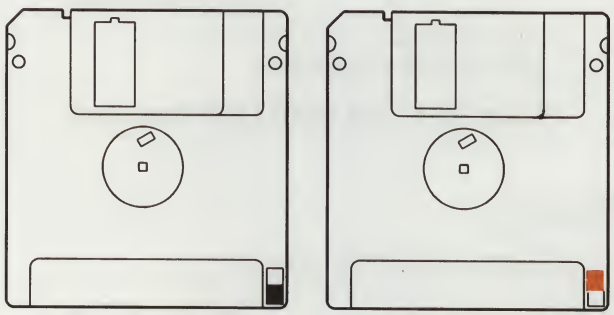


Figure 11-5. Write Protect Source Disk

Verify the Disks Before Copying

1. Insert the source disk into one of your disk drives.
2. Insert the destination disk into the other disk drive.
3. Verify the destination disk by cataloging the disk (include the correct mass storage volume specifier for your second disk drive), for example:

```
CAT ":,701"
```

   If you are copying the entire contents of the source disk, make sure the destination disk does not contain any valuable files since they are destroyed with the copy.

Copying the Entire Contents of One Disk to Another Disk

Make sure you have the correct volume specifiers for both disk drives (using CAT as shown above is the best way to check). Then run the command:

```
COPY "source vs" TO "destination vs"
```

For example,

```
COPY ":,700,0" TO ":,700,1"
```

copies the entire disk at ":,700,0" to the disk at ":,700,1", copying over any information on the disk at ":,700,1".

Copying an Individual File From One Disk to Another Disk

Make sure you have the correct volume specifiers for both disk drives (using CAT as shown above is the best way to check). Then run the command:

```
COPY "source_file:source_vs" TO "destination_file:destination_vs"
```

For example,

```
COPY "Myfile:,700,0" TO "Yourfile:,700,1"
```

copies `Myfile` at ":,700,0" to the disk at ":,700,1", and names it `Yourfile`.

Copying From One Flexible Disk to Another with One Flexible Disk Drive

If you have one flexible disk drive, you can copy individual files from one disk to another. Simply verify that the disks are both formatted, copy from the source to your BASIC/UX hard disk, and use the COPY to copy back to the destination disk, then delete the file off your hard disk.

Prerequisites

Write-protect your source disk (the first disk containing the file or files to be copied).

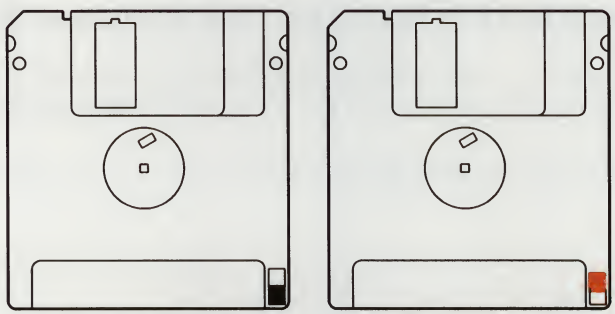


Figure 11-6. Write Protect Source Disk

Copying the Entire Contents of One Disk to Another Disk

1. Create a "memory volume" in your computer's main memory (make sure it is the correct size for the disk you are copying). For details of memory volumes, see the INITIALIZE statement in the *Language Reference*. For example, to create a 270-Kbyte volume:

```
INITIALIZE ":,0"
```

2. Insert the source disk into your flexible disk drive.
3. Copy the entire disk to the memory volume. For example, copy from volume ":,700,1":

```
COPY ":,700,1" TO ":,0"
```

4. Wait a few minutes until `Idle` appears in the lower right corner of the BASIC/UX screen.
5. Remove the source disk and replace it with the destination disk.
6. Use COPY to copy the files from the memory volume to the disk. For example, using the same volumes as above:

```
COPY ":,0" TO ":,700,1"
```

7. Wait a few minutes until `Idle` appears in the lower right corner of the BASIC/UX screen.
8. Verfiy the copy by using CAT:

```
CAT ":,700,1"
```

9. Delete the files from the memory volume. For example,

```
PURGE "FILE:,0"
```

deletes FILE. You can also reinitialize the memory volume as in Step 1 above.

Copying From One Flexible Disk to Another with One Flexible Disk Drive (Continued)

Copying an Individual File From One Disk to Another Disk

1. Make sure you have the correct volume specifiers for the source disk drive (using CAT as shown above is the best way to check).
2. Create a temporary directory on your hard disk to serve as a temporary holding area: `CREATE DIR "temp.copy"` (you can use any name you wish as long as it doesn't already exist).
3. Move to the temporary directory: `MSI "temp.copy"`
4. Run the command:

   `COPY "`*source_file*`:`*source_msvs*`" TO "`*copy_file*`"`

   For example,

   ```
   COPY "Myfile:,700,0" TO "Copyfile"
   ```

   copies `Myfile` at ":,700,0" to the current directory and calls it `Copyfile`.
5. Remove the source disk and insert the destination disk.
6. Run the command:

   `COPY "`*copy_file*`" TO "`*destination_file*`:`*destination_msvs*`"`

   For example,

   ```
   COPY "Copyfile" TO "Yourfile:,700,0"
   ```

   copies `Copyfile` in the current directory to the disk at `:,700,0` and calls it `Yourfile`.
7. Delete the file in `temp.copy` (for example, `PURGE "Copyfile"`).
8. Delete the temporary directory:
   a. Move to the parent directory: `MSI ".."`
   b. Delete the temporary directory: `PURGE "temp.copy"`

Copying Files From an SRM File System

Prerequisites

See your system administrator for your SRM volume specifier.

Using COPY to Copy Files From an SRM File System

1. Verfiy the SRM file system by cataloging it. For example, for a file in the `/USERS` directory on an SRM system at select code 21, node 1:

```
CAT "/USERS:REMOTE 21,1"
```

2. Copy the file. For example, copy the file `Myfile`:

```
COPY "/USERS/Myfile:REMOTE 21,1" TO "Myfile"
```

Notes

A

ITF and Terminal Keyboard Reference

Terminal keyboards that are not ITF keyboards, may have different keys than those used with BASIC/UX. See "Terminal Keyboard Reference."

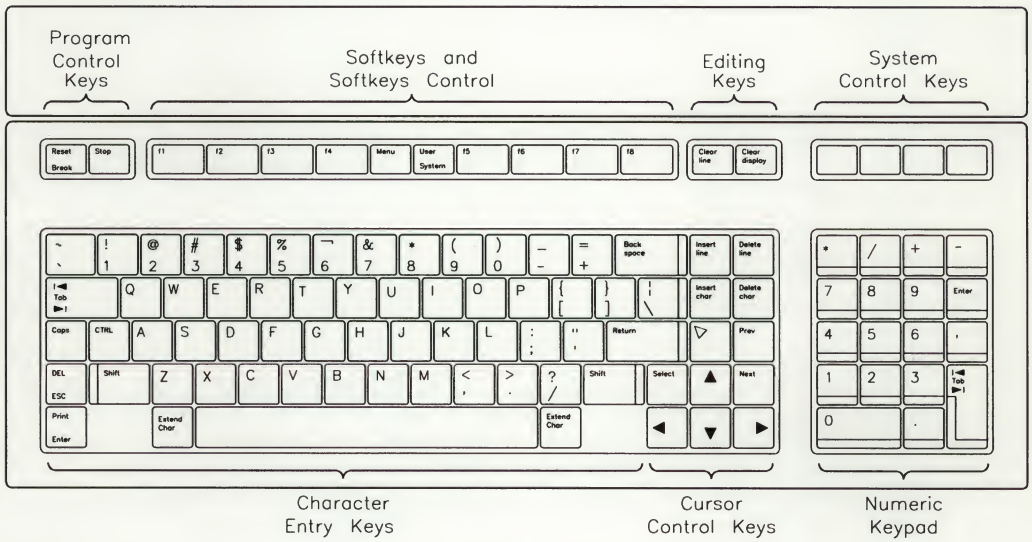


Figure A–1. ITF Keyboard

This chapter provides a handy reference guide to key definitions.

- Keep in mind that other system programs may define the keys differently.
- The **cursor** is the blinking underline pointing to a location on the screen. (If you have an HP 98700 Graphics Display station, HP 98548A, HP 98549A, or HP 98550 display, the cursor does not blink.)
- To clear the computer of previously defined keys, type:

```
SCRATCH
```

Using the BASIC ITF Keyboard Overlays

Two keyboard overlays designed for the ITF keyboard are included with your system. Place the overlays on the keyboard as shown below:

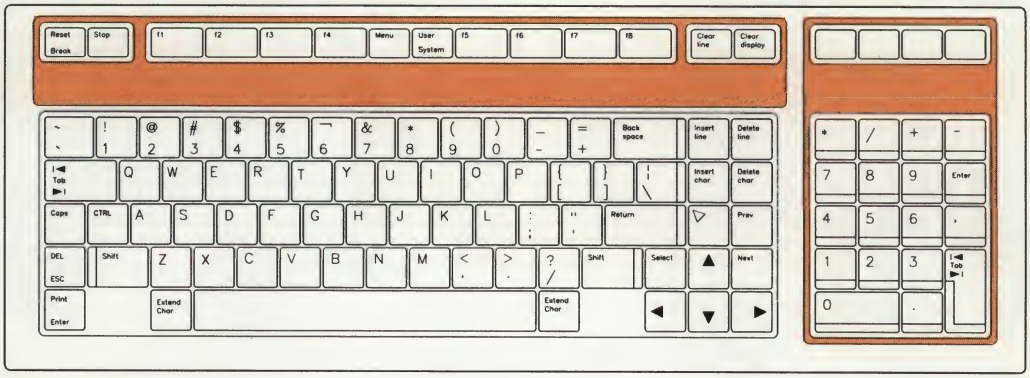


Figure A–2. BASIC Keyboard Overlays

Using Character Entry Keys

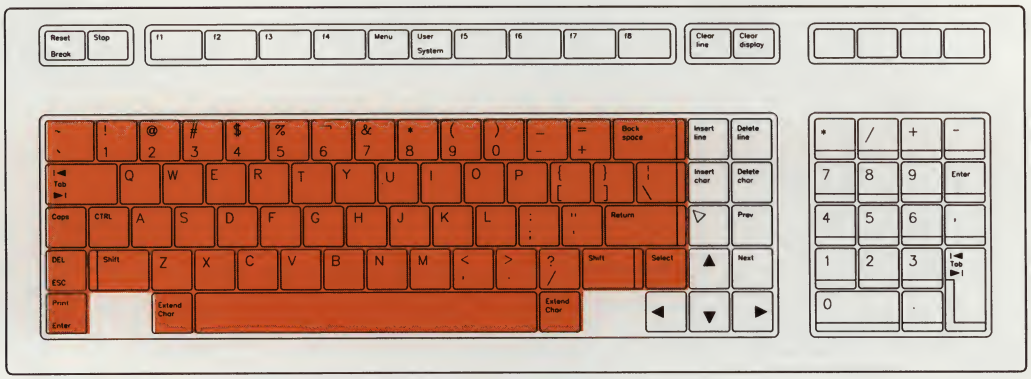


Figure A-3. Keys For Entering Characters

The character entry keys are arranged like a typewriter, but have some added features.

Caps

The [Caps] key sets the unshifted keyboard to either upper-case (which is the default after BASIC is booted) or lower-case (normal typewriter operation). The computer displays which mode the computer is in when you press the [Caps] key.

Type a few words, then press [Caps] and continue typing. Notice the case change. Press [Shift]-[Clear line] when finished.

Shift

You can enter standard upper-case and lower-case letters, using the [Shift] key to access the alternate case.

Type a few words, pressing [Shift] to change the case of the first letter of each word. Now press [Caps] and continue typing. Notice that the alternate case accessed by [Shift] depends on the setting of [Caps]. Press [Shift]-[Clear line] when finished.

Return

The Return key has three functions:

- When a running program prompts you for data, respond by typing the requested data and then pressing Return. This signals the program that you have provided the data and that it can resume execution.
- When typing in lines of a program, the Return key is used to store each line of program code.
- After typing in a command, the Return key causes the command to be executed.

Type EDIT and press Return. Notice the number 10 now displayed on the screen—this is the line number of the first line of a BASIC program. The computer is waiting for you to type in the line. Type:

```
!FIRST LINE
```

and press Return. Notice that the computer accepts the statement as a program line and displays 20 in preparation for the next one. Press Stop when finished.

Enter

Pressing Enter is the same as pressing the Return key.

Print

Pressing Print (Shift-Enter) prints a complete copy of the alpha display on the default printer. The shifted version of the key directly above the / key in the numeric keypad (labeled Dump Alpha on the overlay) performs the same function.

Extend char

When pressed along with another key, this key allows you to generate the rest of the full 256-bit character set from the main typewriter section on Standard and European keyboards (see illustration). On a Katakana keyboard, the "Roman" and "Katakana" keys select the other character sets. To get Katakana characters 161 through 254 on a medium-resolution Series 300 screen, you must load the LEX language extension binary.

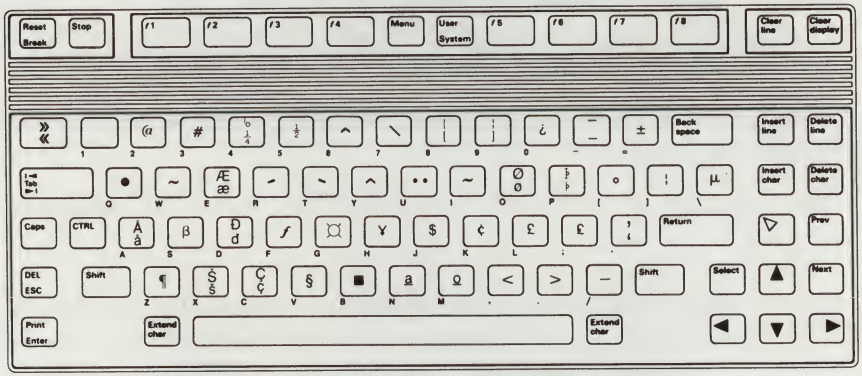


Figure A–4. Extended Character Set

Tab

The Tab key moves the cursor forward to preset tabs. Pressing Shift-Tab moves the cursor backward to preset tabs.

Before Tab can be used, a tab must be set. Tabs are set and cleared with System menu softkeys. The Tab key is demonstrated along with the Set Tab/Clr Tab softkey under "System Softkeys" later in this chapter.

CTRL

The CTRL (control) key works like Shift to access a set of standard control characters, such as line-feed and form-feed. These characters are useful to the programmer for controlling some devices and for communicating with other computers. You probably won't need them when running programs. The available control characters are listed in the *BASIC Language Reference* in the "Useful Tables" appendix.

Select

The Select key beeps but performs no function unless it is program-defined.

Using Cursor-Control Keys

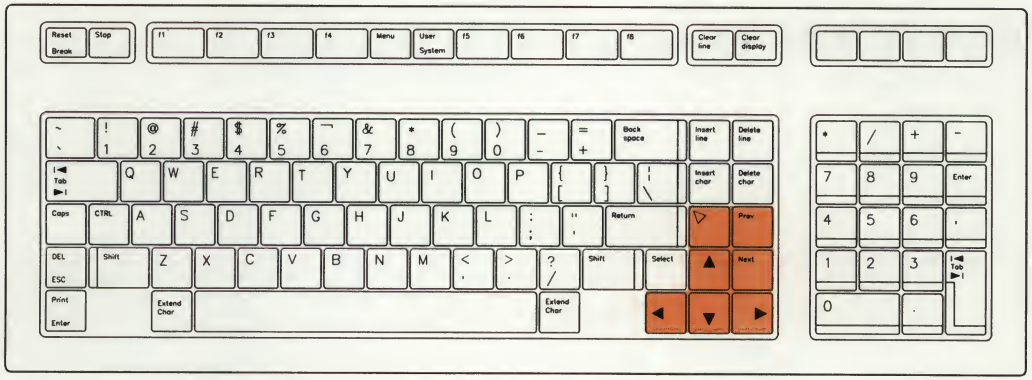


Figure A–5. Keys to Control the Cursor

Cursor-control keys move the display cursor.

- [▲] and [▼] scroll lines in the output area up and down.
- Shifted, the keys allow you to "jump" to the top and bottom of the output area.
- [►] and [◄] moves the cursor horizontally along a line.
- Shifted, they allow you to "jump" to the left and right limits of a line.
- [Back space] works just like the [◄] key.
- The unshifted [➤] positions the cursor at the beginning of the page.
- The shifted [➤] places the cursor at the beginning of the first empty line in the display (scrolls up if necessary). In edit mode, pressing this key (shifted or unshifted) causes the computer to beep.
- In normal mode, [Prev] causes the display to scroll down one page.
- [Next] causes the display to scroll up one page.
- In edit mode, [Next] and [Prev] move the display one-half page.

Using Cursor-Control Keys (Continued)

Testing the Home Key

To test the operation of [↖],

1. Press [Clear display]
2. Type:

```
PRINT "SOMETHING" [Return]
```

3. Repeat steps 1 and 2 twice.

   Your screen should show output like this:

```
SOMETHING
SOMETHING
SOMETHING
```

4. Press the [↖] key (unshifted).
5. Type `PRINT "ANY " and press` [Return]. Your display should look like this:

```
ANY THING
SOMETHING
SOMETHING
```

6. Press [Clear display].

Testing the Horizontal Movement Keys

To test the cursor's horizontal movement,

1. Type a few words
2. Press the shifted and unshifted [◄] and [►] keys. (Notice that the cursor cannot be moved beyond the characters you type.)
3. Press [Shift]-[Clear line] when finished.

Testing the Vertical Movement Keys

To test the cursor's vertical movement,

1. Type: EDIT [Return]
2. Type the following lines, pressing [Return] after each line:

```
10 !FIRST LINE
20 !SECOND LINE
30 !THIRD LINE
```

   Try out the shifted and unshifted [▲], [▼], and [▶] keys. Then try the [Prev] and [Next] keys.

   When you're done,

   a. Press [Stop] to exit.
   b. Type SCRATCH [Return] to clear memory.

Using the Numeric Keypad

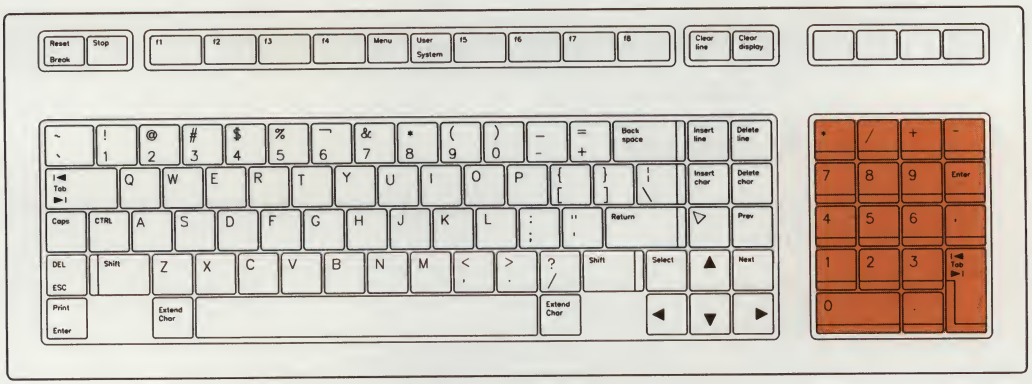


Figure A–6. Numeric Keypad Keys

The numeric keypad provides a convenient way to enter numbers and perform arithmetic operations. Type in the arithmetic expression you want to evaluate, then press [Enter]. The result is displayed in the lower-left corner of the screen.

- [Enter] performs the same function as the [Return] key.
- [Tab] on the numeric keypad functions like [Tab] in the character entry area.
- Shifted versions of [*], [/], [+], and [-] are E, (,), and ^, respectively (see labels on the overlay). The shifted versions are also available in the character entry area.

Type in the following problem using the numeric keypad:

```
(26+14)/4
```

Now press [Enter] to perform the calculation. The answer, 10, is displayed in the lower-left corner of the screen.

Using the Editing Keys

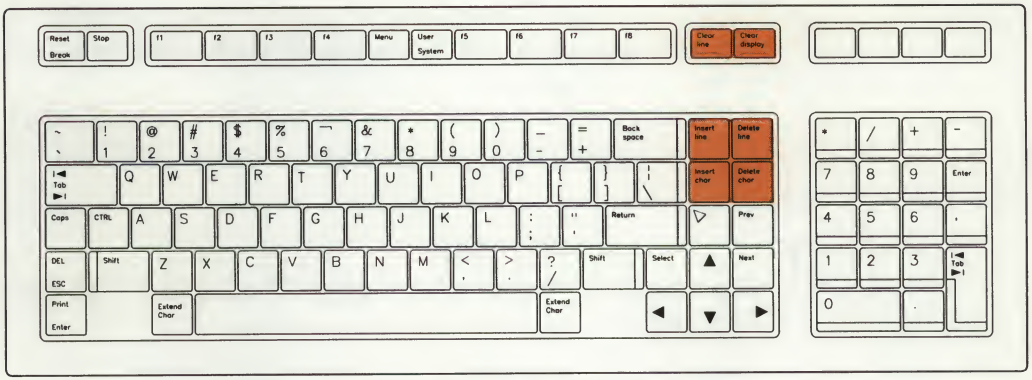


Figure A–7. Edit Keys

Insert line

Inserts a new line above the cursor's current position (edit mode only). Example:

1. Type: `EDIT` Return
2. Type in this line (if not already there): `10 !FIRST LINE`
3. With the cursor somewhere on line 10, press Insert line
   Notice a new number (1) is inserted before line 10.
4. Type `!LINE 1` Return
5. Press Stop when finished.

Delete line

Deletes the line containing the cursor (edit mode only). Example:

1. Type `EDIT` Return
2. Position the cursor to the line: `10 !FIRST LINE`
3. Press Delete line (The line is removed).
4. To restore it, press the key directly above * on the Numeric Keypad (labeled `Recall` on the overlay).
5. Press Return to enter it into the program.
6. Press Stop to exit edit mode.

Using the Editing Keys (continued)

Insert char

Insert char sets insert mode, allowing you to insert characters to the left of the cursor. Press the key a second time to cancel insert mode. Example:

1. Carefully type the following line exactly as shown (while in edit mode) and notice the space between TEST and the period:

   ```
   THIS IS A TEST .
   ```

2. Position the cursor under the period and press Insert char
3. Type:

   ```
   OF INSERT MODE
   ```

4. Press Insert char again. The line should now look like this:

   ```
   THIS IS A TEST OF INSERT MODE.
   ```

   The new characters were inserted to the left of the period.
5. Press Shift-Clear line when finished.

Delete char

Delete char deletes the character at the cursor's position. Type a few words and experiment with Delete char, positioning the cursor at various places on the line. Notice that if you hold the key down, characters are deleted until you release it. Delete all of the characters you typed.

Clear line

- Unshifted-Clear line (labeled Clr—End on the overlay) clears from the current cursor position to the end of the line.
- Shift-Clear line (labeled Clr Ln on the overlay) clears the keyboard line and message/results line.

Type in a few words and use the ◄ key to position the cursor in the middle of the line. Press unshifted-Clear line to clear to the end of the line. Press Shift-Clear line to clear the rest of the line.

Clear display

Either the shifted or unshifted version of Clear display clears the entire alpha screen. Example:

1. Type the following BASIC command (not in EDIT mode):

```
PRINT "PUT THIS MESSAGE IN THE OUTPUT
AREA."
```

2. Press Return to execute it.
3. Press the key directly above * in the Numeric Keypad (labeled Recall on the overlay) to recall the command, and press Return again.
4. Repeat this several times to fill the screen with messages.
5. Press Clear display to erase all lines at once.

Program Control Keys

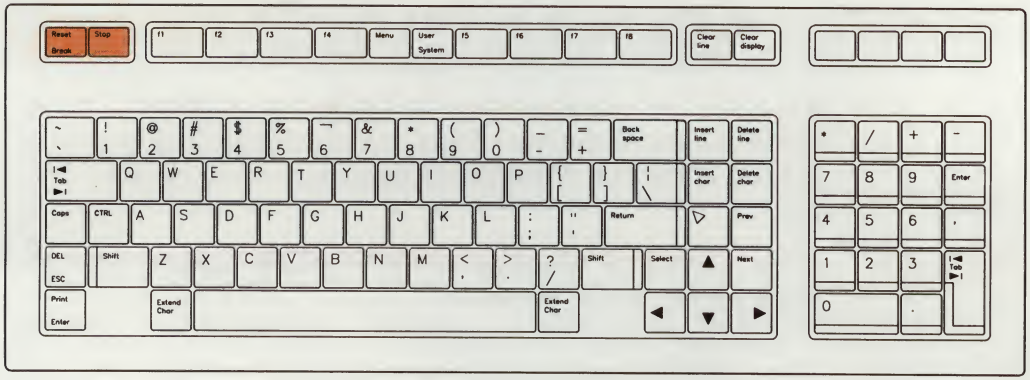


Figure A–8. Keys that Control Your Programs

The following keys allow you to control execution of the program stored in the computer's memory.

Stop

Unshifted-Stop (labeled Pause on the overlay) *pauses* program execution after the current line. Pressing Continue (unshifted f2 in the System menu) resumes program execution from the point where it was paused.

Shift-Stop (labeled Stop on the overlay) *stops* program execution after the current line. To restart the program, press RUN (unshifted f3 in the System menu).

Break

Break (labeled Clr I/O on the overlay) pauses program execution when the computer is performing or trying to perform an I/O operation. Press Break instead of unshifted-Stop when the computer is hung up on an I/O operation, since unshifted-Stop works only after the computer finishes the current program line. Break cancels the I/O operation and pauses the program at the current line.

Reset

Reset (Shift-Break) pauses program execution immediately without erasing the program from memory. The `BASIC Reset` message indicates the computer is ready for your command.

System Control Keys

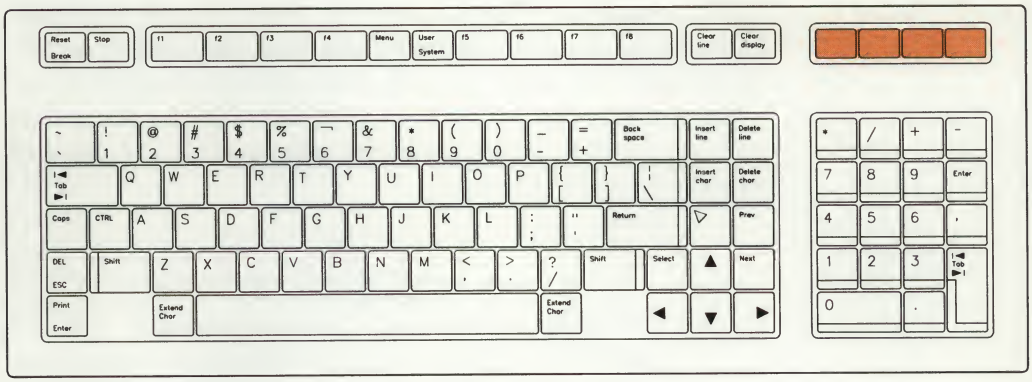


Figure A-9. Unlabeled Control Keys

Four unlabeled keys directly above the numeric keypad control various system functions related to the display, printer, and editing operations. Most of these keys execute their functions immediately, as the key is pressed.

To easily identify the keys in the following description, we'll use this convention:

- Key 1—Above the [*] key (labeled `Recall` on the overlay).
- Key 2—Above the [/] key (labeled `Alpha/Dump Alpha` on the overlay).
- Key 3—Above the [+] key (labeled `Graphics/Dump Graph` on the overlay).
- Key 4—Above the [-] key (labeled `RES` on the overlay).

System Control Keys (Continued)

Key 1--Recall

Unshifted-Key 1 (`Recall`) recalls the last line that you entered, executed, or deleted. Several previous lines can be recalled this way. Recall is particularly handy to use when you mistype a line. Instead of retyping the entire line, you can recall it, edit it using the editing keys, and enter or execute it again. Example:

1. Type:

   `PRINT "1"` [Return]
2. Press Key 1 to recall the print statement.
3. Edit the statement to print the number 2 by positioning the cursor under the 1 and typing [2].
4. Press [Return] again.
5. Press Key 1 several times to see all of the statements it remembers.
6. Press [Clear display] when finished.

[Shift]-Key 1 moves forward through the recall stack.

[f8] in the System menu performs the same recall function as Key 1.

Key 2--Alpha/Dump Alpha

Pressing unshifted-Key 2 (Alpha) once turns on the alphanumeric display. Pressing it the second time turns off the graphics display. This key may perform no function with BASIC/UX.

[Shift]-Key 2 (Dump Alpha) prints a complete copy of the alpha display on the default printer. The Dump Alpha function is also executed by [Print].

`Key 3--Graphics/Dump Graph`

Pressing unshifted-Key 3 (Graphics) once turns on the graphics display. Pressing it the second time turns off the alphanumeric display. This key may perform no function in BASIC/UX.

Shift-Key 3 (Dump Graph) prints a complete copy of the graphics display on the default printer. The combined alpha and graphics display is printed.

`Key 4--RES`

Key 4 (RES) either shifted or unshifted returns the result of the last arithmetic expression that was executed. Example:

1. Press Shift-Clear line
2. Type:

   23+45 Return

   The result, 68, is displayed in the lower-left corner of the screen.
3. To add 123 to this value, press Key 4 and type:

   +123 Return

   The new result, 191, is now displayed.
4. Press Shift-Clear line when finished.

Softkeys and Softkey Control

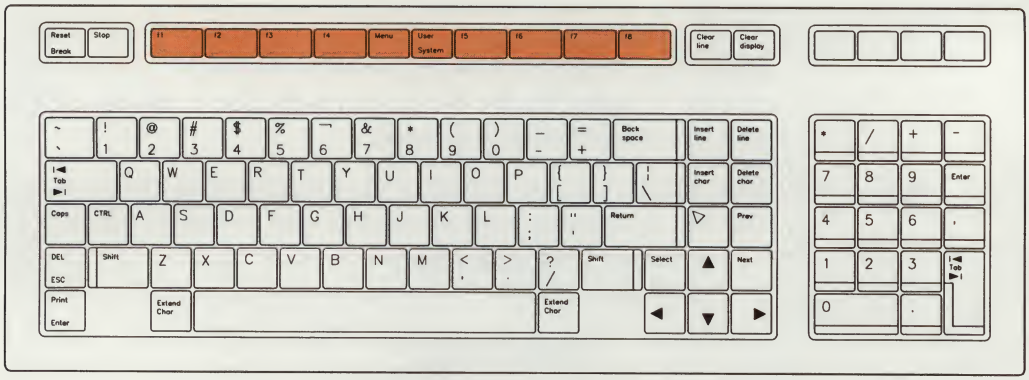


Figure A–10. System Softkeys

There are eight softkeys (labeled f1 through f8 and two keys that control the definitions of the softkeys (Menu and System)).

When BASIC is booted, the softkeys default to System mode. System softkeys are defined following control key definitions. In addition to the System mode, there are also three User modes: User 1, User 2, and User 3. See Chapter 8 for how to change softkey definitions.

Softkey Control Keys

System

Unshifted-System causes softkeys to assume System mode. The System menu is displayed, *if* the Menu key is toggled to the "on" position.

User

User (Shift-System) puts the softkeys in User mode. A User menu is displayed *if* the Menu key is toggled to the "on" position (see Chapter 8 for details).

Menu

Unshifted-Menu toggles the softkey labels—turns them on if they're off and turns them off if they're on.

Shift-Menu increments User mode and menu *if* User mode is "on".

System Softkeys

The following paragraphs define the eight System softkeys.

Step

Step (unshifted-f1) allows you to execute one program line at a time. This is particularly useful for debugging (fixing) programs.

Continue

Continue (unshifted-f2) resumes program execution from the point where it was paused (by an unshifted-Stop).

RUN

RUN (unshifted-f3) starts a program running from the beginning.

Print All

Print All (unshifted-f4) turns the printall mode on and off, copying keyboard operations and displayed error messages to a printall device. Press Print All once to set printall "on" and again to set printall "off". An asterisk (*) next to `All` indicates printall is "on".

The display's output area is the default printall device at powerup. *BASIC Programming Techniques* explains how to select other printall devices.

Example:

1. Press Print All to turn on printall mode.
2. Type in the following command:

```
PRINT "THIS IS A KEYBOARD OPERATION"
```
Return

Both the PRINT command and the message itself are displayed on the screen, which is the default printall device.

3. Type:

```
THIS WILL CAUSE AN ERROR [Return]
```

   Because this is not an executable BASIC statement, an error message is displayed both at the bottom of the screen and in the printall area. A log is produced of all commands typed and executed at the keyboard, along with any error messages.
4. Press [Clear display] to clear the display.
5. Press [Print All] to turn off printall mode.

[Set Tab/Clr Tab]

[Set Tab] (unshifted-[f5]) sets a tab at the cursor's current position. Tabs remain in effect until cleared by either [Clr Tab] or the SCRATCH A statement (see "Clearing Memory" in Chapter 7).

[Clr Tab] ([Shift]-[f5]) clears a tab previously set at the cursor's position.

Example:

1. Press the space bar to move the cursor forward a few spaces.
2. Press [Set Tab]
3. Move the cursor back several spaces using [◄], then press [Tab].
4. Move the cursor forward several more spaces with the space bar, then press [Shift]-[Tab].
5. To clear the tab, move the cursor to the unwanted tab position and press [Clr Tab].
6. Press [Shift]-[Clear line] when finished.

Display Fctns

Display Fctns (unshifted-f6) sets the display-functions mode, allowing you to see special control characters (e.g., form-feed, carriage return) on the screen. Pressing this key a second time cancels the display-functions mode. An asterisk (*) appears next to `Fctns` indicating that display-functions mode is "on".

Example:

1. Type the following line:

   ```
   PRINT "DISPLAY-FUNCTIONS MODE OFF"
   ```
   Return

   Notice the display at the top of the screen.
2. Press Recall (unshifted-f8) to recall the line, and edit it to read:

   ```
   PRINT "DISPLAY-FUNCTIONS MODE ON"
   ```
3. Press Display Fctns
4. Press Return. Notice that the carriage return (CR) and line-feed (LF) control characters are now displayed.
5. Press Display Fctns again to exit display-functions mode.
6. Press Clear display when finished.

Any char

Any char (unshifted-f7) is used to find any ASCII character. Example:

1. Press Any char. The following message appears above the menu:

   ```
   Enter 3 digits, 000 to 255
   ```
2. Enter a three-digit number from 000 through 255 representing the decimal equivalent of an ASCII character. The computer automatically displays the character on the screen. For a list of characters and their equivalent decimal values, see the US ASCII Character Codes table in the "Useful Tables" appendix of the *BASIC Language Reference*.

3. Press [Any char], then type 65 which is the decimal equivalent of "A". The display line now displays "A".
4. Press [Shift]-[Clear line] to erase it.

[Recall]

The [Recall] softkey (unshifted-[f8]) acts just like System Control Key 1 (described earlier) recalls the last line you entered, executed, or deleted. Several previous lines can be recalled this way. [Recall] is particularly handy to use when you mistype a line. Instead of retyping the entire line, you can recall it, edit it using the editing keys, and enter or execute it again. Example:

1. Type:

```
PRINT "1" [Return]
```

   to print the number 1 on the screen.
2. Press [Recall] to recall the PRINT statement.
3. Edit the statement to print the number 2 by positioning the cursor under the 1 and typing [2] over it.
4. Press [Return] again.
5. Press [Recall] several times to see all of the statements it remembers. Note that [Recall] goes backward through the queue.

[Shift]-[f8] allows you to cycle forward through the queue until the last line entered, executed, or deleted is displayed. In the previous exercise you pressed unshifted-[f8] several times, cycling backward through the queue. Now press [Shift]-[f8] several times to cycle forward through the queue until the last line is displayed.

Terminal Keyboard Reference

If you do not have an ITF keyboard, some keys may operate differently than those referenced in this document. In general, only the alphanumeric keys are guaranteed to work properly. The following describes

- which terminal types are supported
- what keys on the terminal can be used to perform the same functions as those on the ITF keyboard
- what device to input graphics from terminals.

Supported Terminal Types

The table below lists the terminal types supported on BASIC/UX. The HP Part Number is generally the same as the terminal type. The following terminal types without the `hp` prefix are also supported.

Table A-1. Terminal Types Supported

hp2393	hp2623
hp2393	hp2625
hp2394	hp2627
hp2397	hp2628
hp2622	hp150
	hp262x

Mapping Terminal Keys to ITF Keyboard Keys

In order to perform the same operation as some of the ITF keyboard keys on a terminal, or some of the BASIC Workstation keys (such as [Recall]),use the following mappings.

Table A-2. Terminal Key Mappings

BASIC System Key	Key Mapping
[Reset]	[CTRL]-[r]
[Clear I/O]	[CTRL]-[c]
[Clear line]	[CTRL]-[e]
[Recall]	[CTRL]-[l]
[Pause]	[CTRL]-[p]

Also, you must note the following:

- [Break], [Reset], [aids], [user] keys, and [modes] keys are not recognized by BASIC/UX (they are local to the terminal).
- Keypad keys on terminals are not recognized by BASIC/UX.
- Shifted non-alphanumeric keys are not recognized by BASIC/UX.

Some Hints While Using Terminals

- There is no [Stop] key. Therefore, to stop a program, type

  STOP [Return]

  (to exit the EDIT mode, use the [Clear Display] or [Clear Line] keys).

- You cannot cycle through the system softkey menus with softkey control keys. Use the SYSTEM KEYS, USER *n* KEYS, or CONTROL KBD,2 statements (see the *BASIC Language Reference*).

- There is no [RECALL] key. Use [Ctrl]-[I] or cycle through the system softkeys for the "RECALL" softkey.

Inputting Graphics from Terminals

Only the keyboard arrow keys are supported as the graphics input device on terminals. See the "Interactive Graphics and Graphics Input" chapter in the *Graphics Techniques* manual for details.

Notes

B

rmb Command Reference

This appendix includes the *HP-UX Reference* page for the `rmb` command.

Notes

NAME

rmb, rmbhil, rmbkbd, rmbtmr, rmbxfr – HP BASIC interpreter (HP BASIC/UX)

SYNOPSIS

rmb [*options*] [*autostart file*]

DESCRIPTION

This command invokes the HP BASIC/UX interpreter which can be used to execute BASIC commands or run BASIC programs.

The BASIC **EXECUTE** command is used to temporarily exit the BASIC environment and spawn a new Bourne shell, from which any number of HP-UX commands can be executed. Ctrl-D terminates the shell and returns to BASIC.

Options

The command line options are:

–b This option causes *rmb* to print a screen on startup which resembles the bootrom screen. This screen contains information about the hardware and software configurations. Several additional fields have been added to the standard bootrom screen:

workspace	size of the *rmb* workspace
swap	information about swap device locations
mounted discs	information about mounted disc locations
HP-UX	information about the HP-UX version

Also, on the interface card lines, information is given as to whether a swap device is on the interface (which precludes BURST I/O), and which device file (if any) is used to access the interface.

–c *file* Specifies that the file *file* should be used for the configuration file rather than **$HOME**/*.rmbrc*.

–e Enables "ignore compatibility errors" mode. This causes incompatible statements from the BASIC workstation to be ignored rather than flagged as errors.

–i Causes *rmb* to run **/usr/lib/rmb/rmbclean** at startup time to reclaim orphaned lockfiles and IPC resources.

–k Causes *rmb* to run **/usr/lib/rmb/rmbconfig -b** at startup time to update the kernel configuration file (**/usr/lib/rmb/rmbbootinfo**).

–l *opt* This option specifies that *rmb* attempt to lock the indicated program segments into memory. This can increase program performance, but at the expense of other programs on the machine. *opt* can be any combination of the characters:

char	segment	size
t	text	2Mb
d	data	200Kb
w	workspace	configurable

The parameter **all** may also be used for *opt* to indicate locking all the segments. Note the approximate size of physical RAM consumed for each segment which is locked. Note also that each of the *rmb* daemons attempts to lock itself into memory. Program locking requires either superuser (root) capabilities, or that the user be a member of the privgrp MLOCK (see *setprivgrp*(1M)).

–n Specifies that the global configuration file **/usr/lib/rmb/rmbrc** is not to be read.

–r *num* This option tells *rmb* to attempt to run at the real-time priority specified by *num*. Valid values for *num* are 0 to 127, where 0 is the highest priority. Note that each of the *rmb* daemons will also attempt to run at this priority. Real-time priority requires either superuser capabilities, or that the user be a member of the privgrp RTPIO (see *setprivgrp*(1M)).

–w *num*[KM] Specifies the *rmb* workspace to be *num* bytes in size. The optional suffix K may be added to represent Kilobytes, or M for Megabytes. The default workspace size is 1Mb.

–x *display* Specifies that *display* is to be used as the **X** windows display server for the *rmb* window.

AUTHOR

rmb was developed by HP

FILES

/dev/rmb/*	location of device files used by rmb
/usr/bin/rmb*	the rmb executable
/usr/bin/rmbbuildc	program for building CSUBs
/usr/include/csubdecl.h	include file for compiling CSUBs
/usr/lib/librmb*.a	rmb library for linking CSUBs
/usr/lib/rmb/.lock/lock*	the resource information lockfile
/usr/lib/rmb/demos/*	demonstration programs and manual examples
/usr/lib/rmb/fonts/*	default bitmapped character fonts
/usr/lib/rmb/ipcclean	the ipcclean program
/usr/lib/rmb/newconfig/*	example configuration files
/usr/lib/rmb/rmbbootinfo	rmb configuration information file
/usr/lib/rmb/rmbclean	the rmbclean script
/usr/lib/rmb/rmbconfig	the rmbconfig program
/usr/lib/rmb/rmbdfile	kernel configuration file scanner
/usr/lib/rmb/rmbhil	the HIL interface daemon
/usr/lib/rmb/rmbkill	program to kill EXECUTE processes after RESET
/usr/lib/rmb/rmbkbd	the keyboard daemon
/usr/lib/rmb/rmbrc	the global configuration file
/usr/lib/rmb/rmbtmr	the timer daemon
/usr/lib/rmb/rmbxfr	the TRANSFER statement daemon
/usr/lib/rmb/utils/*	rmb utilities and CSUBs
$HOME/.rmbrc	the local user configuration file

SEE ALSO

Using the BASIC/UX System, BASIC Language Reference

INTERNATIONAL SUPPORT

rmb: messages

Index

A

B

C

D

E

F

G

H

M

N

O

P

Q

R

S

T

U

V

W

X

HP Part Number
98796-90000
Microfiche No. 98796-99000
Printed in U.S.A. E0988

98796-90600
For Internal Use Only